



BEA eLink

Business Process Option

User's Guide

BEA eLink Business Process Option 1.2
Document Edition 1.2
February 2000

Copyright

Copyright © 2000 BEA Systems, Inc. All Rights Reserved.

Restricted Rights Legend

This software and documentation is subject to and made available only pursuant to the terms of the BEA Systems License Agreement and may be used or copied only in accordance with the terms of that agreement. It is against the law to copy the software except as specifically allowed in the agreement. This document may not, in whole or in part, be copied photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from BEA Systems, Inc.

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the BEA Systems License Agreement and in subparagraph (c)(1) of the Commercial Computer Software-Restricted Rights Clause at FAR 52.227-19; subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, subparagraph (d) of the Commercial Computer Software--Licensing clause at NASA FAR supplement 16-52.227-86; or their equivalent.

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FURTHER, BEA Systems DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE SOFTWARE OR WRITTEN MATERIAL IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

Trademarks or Service Marks

BEA, ObjectBroker, TOP END, and TUXEDO are registered trademarks of BEA Systems, Inc. BEA Builder, BEA Connect, BEA Manager, BEA MessageQ, Jolt and M3 are trademarks of BEA Systems, Inc.

All other company names may be trademarks of the respective companies with which they are associated.

BEA eLink Business Process Option User's Guide

Document Edition	Part Number	Date	Software Version
1.2		February 2000	eLink Business Process Option 1.2

Contents

About this Guide

Who Should Read This Document.....	xi
How This Document Is Organized.....	xi
How to Use This Document.....	xii
Documentation Conventions.....	xii
Related Documentation.....	xiv
eLink Business Process Option Documentation.....	xiv
BEA Publications.....	xv
Other Publications.....	xv
Contact Information.....	xv
Customer Support.....	xvi

1. eLink Overview

How eLink Works.....	1-1
Transferring Funds: Traditional Banking Scenarios.....	1-2
Scenario 1: Withdrawal from Account in USA.....	1-2
Scenario 2: Deposit to Account in Switzerland.....	1-3
Scenario 3: Funds Transfer (USA to Switzerland).....	1-4
Transferring Funds: Electronic Banking Scenario.....	1-5
Data Processing in eLink.....	1-9

2. Business Process Design Overview

Overview.....	2-1
Business Interface Window.....	2-1
Business Process Window.....	2-2
About Business Process Option users.....	2-2
About the Process Flow Model.....	2-3

Tasks.....	2-3
Roles and pools.....	2-4
Process instance life cycle	2-4
Process definitions	2-5
Flat definitions.....	2-5
Conditional definitions	2-6
Task dependencies	2-8
Kinds of task states	2-9
About attributes	2-10
Distinguishing process and task attributes	2-12
About process ownership.....	2-12
Creating Business Processes: An Overview	2-13

3. Specifying Business Service Contracts

Overview	3-1
How It Works	3-2
Key Components	3-3
Business Interface Window	3-3
Accessing Contract Repository Objects	3-4
Contract Repository Tree View.....	3-4
Refreshing the Contract Repository Tree View	3-8
Toolbar Buttons	3-8
Keyboard & Menu Shortcuts.....	3-10
Usage	3-12
Logging on and Getting Started	3-13
Starting the Process Design Assistant	3-14
Logging On to the Process Design Assistant	3-15
Setting Logon Option Defaults	3-15
Logging On	3-16
Importing FML Files	3-17
FML Import Instructions	3-20
Loading Externally Described Interfaces	3-21
Instructions for Loading Externally Described Interfaces	3-21
Creating New Objects.....	3-22
Format Rules for Repository Objects.....	3-22

Adding Elements.....	3-22
Adding Structs.....	3-26
Adding Operations.....	3-28
Adding Interfaces.....	3-31
Adding Modules.....	3-33
Modifying Objects.....	3-34
Modifying Elements.....	3-35
Modifying Structs.....	3-37
Modifying Parameters.....	3-38
Modifying Operations.....	3-41
Modifying Interfaces.....	3-43
Modifying Modules.....	3-45
Deleting Objects From the Repository.....	3-48
Deleting Instructions.....	3-48
Exporting or Unexporting an Operation.....	3-50
Export Instructions.....	3-51
Unexport Instructions.....	3-52
Testing an Operation.....	3-53
Guidelines for Inputting Data.....	3-54
Test Operation Instructions.....	3-54
Logging Off and Exiting.....	3-55
Logoff/Exit Instructions.....	3-55
Troubleshooting.....	3-56
Alert Window Instructions.....	3-56
Process Design Assistant and ATMI Mappings.....	3-57

4. Designing Business Processes

Overview.....	4-1
How It Works.....	4-1
Usage.....	4-2
Generating a Palette.....	4-2
Instructions for Generating a Palette.....	4-3
Generating a Template.....	4-4
Instructions for Generating a Template.....	4-4
Opening the Business Process Window.....	4-5

Instructions for Opening the Business Process Window	4-5
Confirming Login to the Business Process Database.....	4-6
Using the Business Process Window	4-6
Using the Business Process Window.....	4-8
Getting started with the Business Process Window	4-8
Refreshing the Design Pad display.....	4-10
Printing from the Design Pad	4-11
Processes are automatically saved.....	4-11
Defining and changing process structure	4-12
Navigating the process structure	4-13
Instructions for Navigating from the Hierarchy Pane	4-14
Instructions for Hiding a Task's Children in the Hierarchy Pane	4-15
Instructions for Navigating from the Task Pane	4-15
Instructions for Navigating from the Property Sheet	4-15
Working with multiple Design Pad windows.....	4-16
Instructions for Working on Different Parts of the Same Process	4-16
Instructions for Working on a Different Process	4-16
Copying tasks between Design Pad windows	4-17
Using drag and drop between Design Pad windows	4-17
Working with Process Definitions.....	4-19
Working with existing process definitions	4-19
Instructions for Preparing to Form a Process Flow.....	4-19
Viewing process attributes	4-21
Summary of built-in process attributes	4-22
Validating process definitions.....	4-22
Working with Tasks.....	4-22
Copying tasks	4-22
Instructions for Copying a Task	4-23
Instructions for Copying a Task with Drag-and-Drop	4-24
Working with task objects in the Design Pad.....	4-25
Selecting and moving tasks.....	4-25
Instructions for Selecting and Deselecting Tasks and Dependencies	4-25
Instructions for Moving a Task	4-25
Instructions for Selecting a Task in Another Layout Mode.....	4-26
Instructions for Moving a Task in Another Layout Mode	4-26

Changing the attribute display	4-26
Naming tasks	4-27
Aligning tasks	4-27
Adding dependencies	4-29
Deleting tasks and dependencies.....	4-30
Customizing task icons.....	4-30
Adding customized icons to the icon library	4-31
Changing task icons	4-31
Specifying attributes	4-34
Instructions for Opening the Property Sheet	4-34
Instructions for Viewing the Attribute Values for a Task.....	4-35
Instructions for Setting an Attribute Value or Modifying an Existing Value	4-35
Instructions for Adding an Attribute to be Used in a Perform Condition.	4-35

5. Making eLink Processes Work

Business Process and Task Parameter Attributes	5-1
Parameter Attributes & Default Assignments.....	5-2
Parameter Assignments Explained.....	5-2
Data Flow Through the Business Process Engine.....	5-3
Data Flow By the Numbers.....	5-4
Working with Conditions	5-8
General form of a PerformCondition or IterateCondition.....	5-9
Attributes in PerformConditions	5-9
Properties in PerformConditions	5-10
Operators used in PerformConditions	5-11
Data types used in PerformConditions.....	5-11
Compound statements	5-12
Grouping statements with parentheses.....	5-12
Using internal attributes	5-13
Summary of built-in task attributes.....	5-14
About attribute values	5-15
Iterating tasks and subtasks	5-16
Examples of PerformConditions	5-16

Adding PerformConditions to tasks	5-17
Instructions for Specifying a PerformCondition	5-17
Instructions for Creating a PerformCondition based on Task Attributes .	
5-18	
Instructions for Creating a PerformCondition with Evaluation	
Expressions	5-22
Instructions for Specifying an IterateCondition	5-23
Instructions for Creating an IterateCondition Based on Task Attributes .	
5-24	
Instructions for Creating an IterateCondition with Evaluation	
Expressions	5-28
Working with Process Instances	5-29
Instructions for Starting a Process	5-30
Managing Processes with the Process Manager	5-30
Opening the Process Manager	5-30
Instructions for Starting the Process Manager	5-31
Completing a process	5-32
Instructions for Forcing a Process to Completion	5-32
Deleting a complete process	5-33
Instructions for Deleting a Completed Process	5-33
Modifying a process instance in progress	5-33
Instructions for Modifying Parts of a Process	5-34
Suspending and resuming a process	5-34
Instructions for Suspending a Process	5-35
Instructions for Resuming a Process	5-35
Viewing the status of a process	5-35
Instructions for Viewing the Status of a Process	5-36
Instructions for Viewing Other Levels of the Process	5-36
Instructions for Changing the Information Displayed for Each Task	5-37
Instructions for Viewing the Details of a Specific Task	5-37
Instructions for Viewing the Properties of a Task	5-38
Using folders to organize the Process Manager	5-38
Instructions for Creating a New Folder	5-39
Finding and adding process definitions to folders	5-39
Instructions for Finding a Process Definition and Adding It to a Folder .	
5-39	

Instructions for Renaming a Folder	5-40
Moving processes between folders	5-41
Deleting processes from personal folders	5-42
Customizing how processes are displayed in folders.....	5-42

A. Interface File (*.IFCE) Reference

Overview	A-1
Loading ATMI Services	A-2
Introduction to the Interface Loader.....	A-2
Syntax of the Interface Files.....	A-2
Guidelines for Using Keywords.....	A-3
Keyword Order in the Interface Loader Data File	A-3
Using Service-Level Keywords and Values	A-4
Using Parameter-Level Keywords and Values	A-5
Troubleshooting	A-6
Interface File Example	A-6
Interface File Explained	A-7

B. MATHAPP Tutorial & Reference

Creating a Simplified Funds Transfer Process Flow: a Tutorial	B-1
Logging On	B-2
Loading an Interface & Adding an Operation.....	B-3
Exporting & Testing Operations	B-5
Generating Palette & Template	B-6
Moving to the Business Process Window	B-6
Setting Up Your Process Flow	B-7
Modifying Tasks and Processes	B-8
Adding Your New Process to the Configuration	B-10
Testing Your Process Flow	B-11
Assigning Parameters to Implement Process Flows	B-12
Root Level Input & Output	B-12
Task Level Input & Output (SubtractInteger).....	B-13
Task Level Input & Output (AddInteger)	B-14
MATHAPP Service Reference.....	B-16

Glossary

Index

About this Guide

This document describes the BEA eLink Business Process Option product and gives instructions for using the tools for linking applications together using the eLink Business Process Option.

Who Should Read This Document

This document is intended for business analysts who are interested in linking applications together into a distributed environment.

How This Document Is Organized

The BEA eLink Business Process Option User's Guide is organized as follows:

- Chapter 1, “eLink Overview,” provides an overview of eLink in its entirety.
- Chapter 2, “Business Process Design Overview,” provides a quick study of the Business Process Option and the procedures for using it. More detailed instructions can be found in later chapters. This chapter includes a flow chart.
- Chapter 3, “Specifying Business Service Contracts,” provides detailed instructions for using the Business Interface Window within the eLink Process Design Assistant to specify Business Service Interfaces.
- Chapter 4, “Designing Business Processes,” provides detailed instructions for using the Business Interface and Business Process Windows to build process flows.
- Chapter 5, “Making eLink Processes Work,” provides instructions for adding conditions to and modifying parameters in tasks to make their processes work.

-
- Appendix A, “Interface File (*.IFCE) Reference,” provides information on the layout of files used to load interface specifications.
 - Appendix B, “MATHAPP Tutorial & Reference,” provides a tutorial for using the eLink Business Process Option with MATHAPP. It also provides a MATHAPP functional reference.
 - Glossary

How to Use This Document

This document, the BEA eLink Business Process Option User’s Guide, is designed primarily as an interlinked HTML document set. A virtual print document in Adobe Acrobat will also be provided.

Documentation Conventions

The following documentation conventions are used throughout this document.

Convention	Item
boldface text	Indicates terms defined in the glossary.
Ctrl+Tab	Indicates that you must press two or more keys simultaneously.
<i>italics</i>	Indicates emphasis or book titles.

Convention	Item
monospace text	<p>Indicates code samples, commands and their options, data structures and their members, data types, directories, and file names and their extensions. Monospace text also indicates text that you must enter from the keyboard.</p> <p><i>Examples:</i></p> <pre>#include <iostream.h> void main () the pointer psz chmod u+w * \tux\data\ap .doc tux.doc BITMAP float</pre>
monospace boldface text	<p>Identifies significant words in code.</p> <p><i>Example:</i></p> <pre>void commit ()</pre>
<i>monospace</i> <i>italic</i> text	<p>Identifies variables in code.</p> <p><i>Example:</i></p> <pre>String <i>expr</i></pre>
UPPERCASE TEXT	<p>Indicates device names, environment variables, and logical operators.</p> <p><i>Examples:</i></p> <pre>LPT1 SIGNON OR</pre>
{ }	<p>Indicates a set of choices in a syntax line. The braces themselves should never be typed.</p>
[]	<p>Indicates optional items in a syntax line. The brackets themselves should never be typed.</p> <p><i>Example:</i></p> <pre>buildobjclient [-v] [-o name] [-f file-list]... [-l file-list]...</pre>
	<p>Separates mutually exclusive choices in a syntax line. The symbol itself should never be typed.</p>

Convention	Item
...	<p>Indicates one of the following in a command line:</p> <ul style="list-style-type: none"> ■ That an argument can be repeated several times in a command line ■ That the statement omits additional optional arguments ■ That you can enter additional parameters, values, or other information <p>The ellipsis itself should never be typed.</p> <p><i>Example:</i></p> <pre>buildobjclient [-v] [-o name] [-f file-list]... [-l file-list]...</pre>
.	<p>Indicates the omission of items from a code example or from a syntax line.</p> <p>The vertical ellipsis itself should never be typed.</p>

Related Documentation

The following sections list the documentation provided with the eLink software, and other publications related to its technology.

eLink Business Process Option Documentation

The eLink Business Process Option information set consists of the following documents:

BEA eLink Business Process Option User's Guide

BEA eLink Business Process Option Release Notes

BEA eLink Business Process Option Operations & Maintenance Guide

BEA eLink Business Process Option Configuration Guide

BEA eLink Business Process Option Functional Specification

Note: The BEA eLink Business Process Option CD set also includes Adobe Acrobat PDF files of all of the online documents. You can use the Adobe Acrobat Reader to print all or a portion of each document.

BEA Publications

The following BEA publications which cover the eLink platform technology in depth are also available:

TUXEDO System 6 Administration Guide

TUXEDO System 6 Administration Guide to the Web-Based GUI

TUXEDO System 6 Reference Manual

Other Publications

For more information about the eLink Platform technology, refer to the following books:

3-Tier Client/Server at Work (Edwards)

The TUXEDO System (Andrade, Carges, Dywer, Felts)

The BEA eLink Business Process Option incorporates 3rd-party process engine technology. The relevant documentation is directly incorporated within the ELink Business Process Option documentation set. While this should provide all the information necessary, the eLink Business Process Engine CD also contains the 3rd-party documentation for reference. Please note that neither this documentation nor the usage it describes are directly supported by BEA Systems, Inc.

Contact Information

The following sections provide information about how to obtain support for the documentation and software.

Customer Support

If you have any questions about this version of the BEA eLink Business Process Option, or if you have problems installing and running the BEA eLink Business Process Option, contact BEA Customer Support through BEA WebSupport at www.beasys.com. You can also contact Customer Support by using the contact information provided on the Customer Support Card, which is included in the product package.

When contacting Customer Support, be prepared to provide the following information:

- Your name, e-mail address, phone number, and fax number
- Your company name and company address
- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

1 eLink Overview

This is an overview of the entire eLink system. The Business Process Option is just one part of eLink, but to understand the Business Process Option, it is important to have an understanding of eLink's big picture. The following topics are covered in this eLink Overview:

- How eLink Works
- Transferring Funds: Traditional Banking Scenarios
- Transferring Funds: Electronic Banking Scenario
- Data Processing in eLink

How eLink Works

eLink integrates enterprise applications. The applications are connected through the eLink Platform, application adapters (software), and data integration. Business processes are directed by the Business Process Option through the eLink platform, data integration (if needed), and adapters, then processed in the application. The type of application and any necessary adapters or data integration is transparent to the users of the Business Process Option. As long as the correct inputs are received by the application in a form that the application can understand, the application executes the transaction.

Transferring Funds: Traditional Banking Scenarios

To explain eLink, we will start in this section of the chapter with very traditional scenarios. In the next section of the chapter, we will show you the eLink analogies to these traditional scenarios. These analogies will help describe the components of eLink and lead up to a description of the entire eLink system.

The processes we use to execute transactions at a bank, in both the traditional and electronic banking scenarios, require that we use certain tools. Understanding the roles of the participants and how they use these tools to execute transactions is the key to understanding eLink. It is also important to note that the type of phones used and the telephone network particulars are matters that are transparent to those executing the process. Just as, the application adapters and eLink platform are transparent to users of eLink.

Scenario 1: Withdrawal from Account in USA

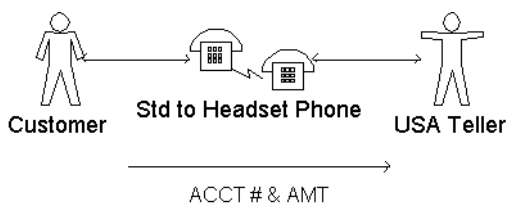
A withdrawal is the simplest and most straight forward transaction, so we will start with a withdrawal scenario.

In this scenario, the customer uses a standard phone to make the call, the teller answers with a headset phone, and they are communicating using a telephonic network.

The following is a procedure for processing a withdrawal:

1. Customer calls bank; teller answers.
2. Customer requests a withdrawal of funds.
3. Customer provides account number and the amount of withdrawal.

Figure 1-1 Customer to Teller Withdrawal Interaction



4. Teller processes withdrawal.

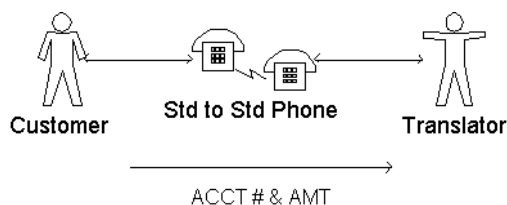
Scenario 2: Deposit to Account in Switzerland

This transaction is more complex because the tellers at the bank branch in Switzerland speak only French and the customer speaks only English. Therefore, a translator is needed to facilitate the transaction.

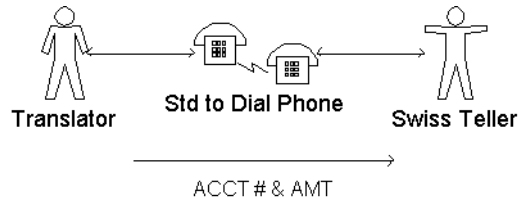
In this scenario, the customer uses a standard phone to make the call, the translator is also using a standard phone, but the teller is using a headset phone, and they are all communicating using the telephone network.

The following is a procedure for processing a deposit at a non-English speaking branch:

1. Customer calls the bank and speaks to a translator.
2. Customer requests deposit of funds providing account number and the amount of deposit.



3. Translator passes on customer's request to teller after translating it to French.



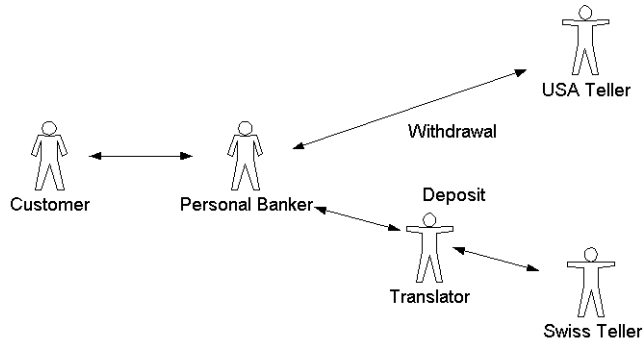
4. Teller processes deposit.

Scenario 3: Funds Transfer (USA to Switzerland)

This transaction is the most complex because of the complexities associated with the language problems, calling offshore bank branches, etc. Because of these complexities, the customer decides to use a Personal Banker. A Personal Banker is a bank transaction specialist who acts as an agent for the customer to execute all of the complex transactions associated with an international funds transfer.

1. The customer calls the bank and speaks to a Personal Banker. The customer is using a standard telephone and the Personal Banker is using a standard phone. The customer requests a transfer of funds from a branch in the USA to a branch in Switzerland by providing account number and the amount of transfer.
2. The Personal Banker contacts the USA teller and requests a withdrawal for the amount of the transfer. The Personal Banker is using a standard telephone and the USA Teller is using a headset phone.
3. The USA teller processes the withdrawal portion of the transfer.
4. Personal banker contacts translator and asks translator to pass on to Swiss Teller a request for a deposit in the amount of transfer. The Personal Banker is using a standard telephone and the Translator is using a standard phone. This is part of the deposit portion of the transfer process.
5. Translator passes on the Personal Banker's request to the teller after translating it to French. The translator is using a standard telephone and the Swiss Teller is using a dial phone.

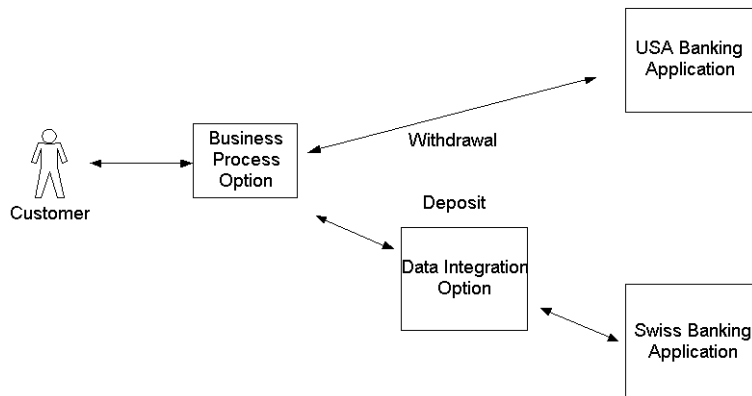
- The teller processes the deposit portion of the transfer. The transfer is completed.



Transferring Funds: Electronic Banking Scenario

This section provides a scenario that is the eLink equivalent to the traditional banking Scenario 3 of the previous section. Notice how similar these drawings are.

Figure 1-2 Electronic Banking Transfer Process



The following table compares the tools used in the traditional banking scenarios above with the electronic banking example below.

Table 1-1 Comparing Tools Used: Traditional & Electronic Banking Scenarios

Traditional Tool	Electronic Tool
US Teller	USA Banking Application
Swiss Teller	Swiss Banking Application
Headset Phone	Adapter for USA Banking Application
Dial Phone	Adapter for Swiss Banking Application
Telephone Network	eLink Platform
Translator	eLink Data Integration Option
Personal Banker	eLink Business Process Option

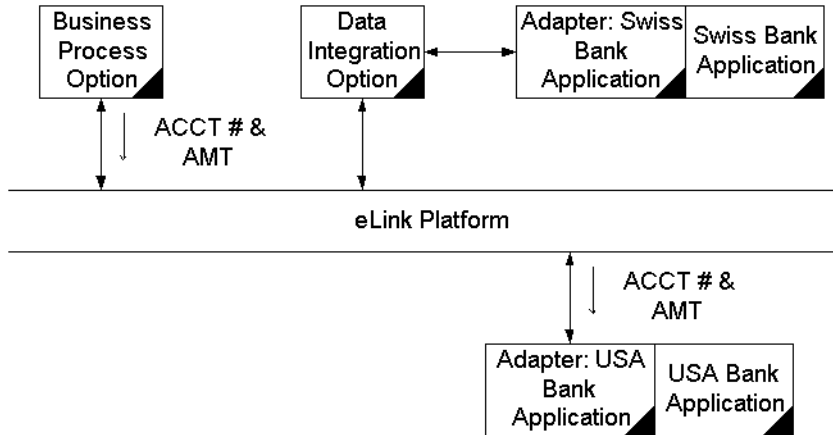
The headset and dial phones are analogous to the application adapters in that they are devices that allow the tellers, which are analogous to the banking applications, to communicate with the other participants in the transaction process through the telephone network (analogous to the eLink platform).

The translator is analogous to the eLink Data Integration Option. The translator translates language data from English to French, then back to English. The Data Integration Option translates data understandable to the eLink platform into data understandable to the bank applications.

Here's the step-by-step scenario procedure:

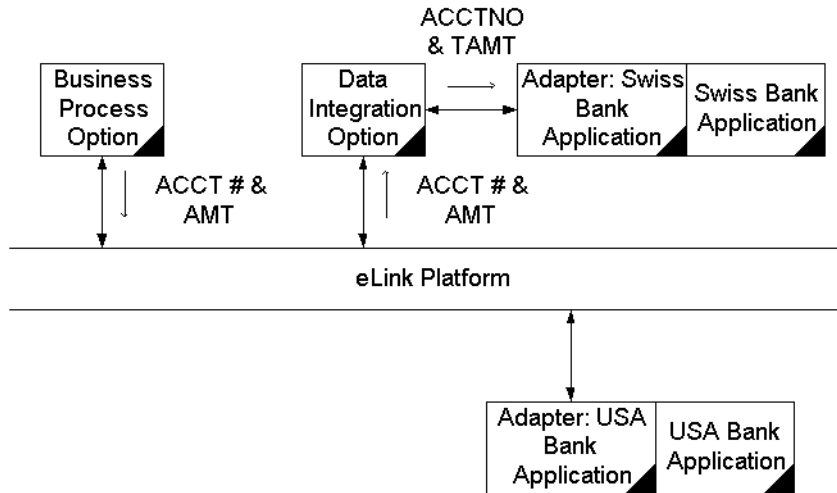
1. Our customer submits a request for transfer of funds through our bank's online banking web page.
2. The eLink Business Process Option (analogous to the Personal Banker in the scenarios above) requests a withdrawal from the USA bank. It sends the customer's account number and transfer amount through the eLink platform (analogous to the telephone network in the scenarios above) and the USA Bank application adapter (analogous to the headset phone in the scenarios above) to the USA Bank application (analogous to the USA Teller in the scenarios above).

Figure 1-3 Withdrawal Using eLink



3. The USA Bank application processes the withdrawal portion of the transfer.
4. The eLink Business Process Option (analogous to the Personal Banker in the scenarios above) requests a deposit to the Swiss bank. It sends the customer's account number and transfer amount through the eLink platform (analogous to the telephone network in the scenarios above), the Data Integration Option (analogous to the translator in the scenarios above), and the Swiss Bank application adapter (analogous to the dial phone in the scenarios above) to the Swiss Bank application (analogous to the Swiss Teller in the scenarios above).

Figure 1-4 Deposit Using eLink



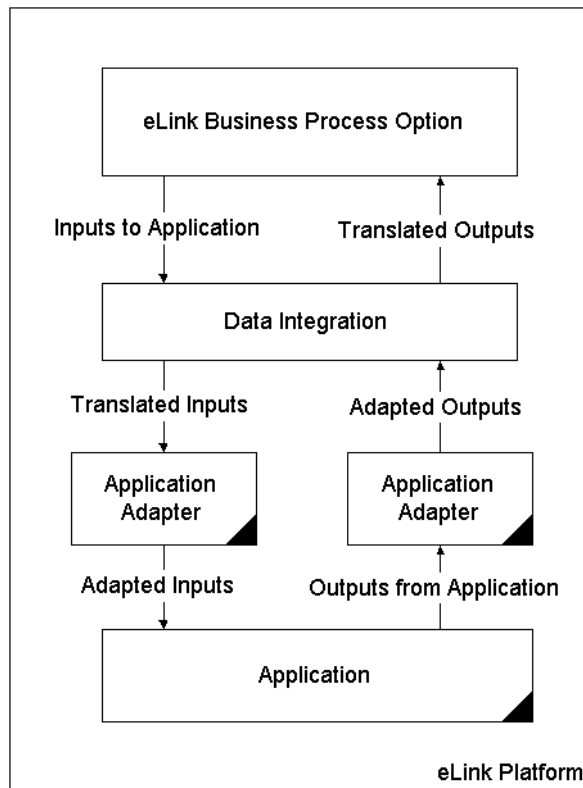
5. The Swiss Bank application processes the deposit portion of the transfer, and the transfer is completed.

In the next section, we will show you how data moves through the eLink system and how it is processed.

Data Processing in eLink

Figure 1-2 is an illustration of an eLink process flow. This section shows you a more discrete example of how data moves through the eLink system and how it is processed. Figure 1-5 illustrates a typical flow of data through eLink.

Figure 1-5 Data Flow Through eLink



The process steps in eLink are called tasks. If the first task is a withdrawal, it corresponds to the banking example in Figure 1-2. Actually, the diagram in Figure 1-5 can apply to all tasks in the process flow, because the data flows through eLink in the same manner regardless of the type of data integration, adapters, or application that processes it.

2 Business Process Design Overview

The following topics are covered in this Business Process Design Overview:

- Overview
- About Business Process Option users
- About the Process Flow Model
- Process definitions

Overview

This chapter is a guide or roadmap to designing business processes using the BEA eLink Business Process Option. It includes a discussion of the different parts of the eLink Process Design Assistant.

Business Interface Window

Use the tools in this window to create interfaces to the applications connected through eLink. The various methods for creating these interfaces are discussed in Chapter 3, “Specifying Business Service Contracts.” The operations contained within these interfaces are converted into tasks. Then, the tasks are used to form process flows (in

the Business Process Window). Only tasks generated from interfaces defined using the Business Interface Window can be used to form process flows that are valid to use in eLink.

Business Process Window

In this window, tasks generated from interface operations are used to form process flows. The various methods for forming these process flows are discussed in Chapter 4, “Designing Business Processes,” and Chapter 5, “Making eLink Processes Work.”

About Business Process Option users

People who use the Business Process Option and associated tools fulfill several roles. Each role uses specific applications or utilities

Table 2-1 Users and Tools

Role of Business Process Option user	Responsibilities	Business Process Option tool
Application Interface Designer	creates interfaces to applications that are connected to eLink	Process Design Assistant (Business Interface Window) Interface File Loader FML Import Utility
Process designer	creates process definitions by capturing an existing business process or by redesigning a business process	Process Design Assistant (Business Process Window) Batch Registry
Process manager	modifies, completes, and monitors processes	Process Manager Batch Registry

Table 2-1 Users and Tools

Administrator	oversees the running of the eLink Business Process Engine and database	Batch Registry Process Manager
---------------	--	-----------------------------------

This document describes all the tools shown in Table 2-1, except for the Batch Registry utility, which is described in the *eLink Business Process Option Administration Guide*.

About the Process Flow Model

The following concepts are key to understanding our process flow model.

A *process* (also known as a *job*) is a structured activity involving multiple steps.

A *process instance* is a business activity in action. A process instance is based on a *process definition* (also known as a *template*), and one or more instances of a process definition can be active at any time. A process instance is performed by one or more workers. In the Business Process Option, these workers are eLink software agents.

The performance of a process requires:

- Tasks
- Roles
- Pools

Tasks

Tasks in a process are organized into a hierarchy. For example, the task of sending an insurance policy renewal form can be divided into three subtasks:

- Compute new premium
- Notify local office
- Mail form to customer

When a process is run, tasks go through one of these states:

- **Waiting** means that the Task is not yet Ready.
- **In Process** means that a worker has acquired the Task.
- **Done** means that a worker has completed the Task.
- **Skipped** means that the Task will never be Ready, but tasks that depend on the Skipped task can become Ready.
- **Activated** means that the Task has been automatically started without being associated with a worker. This state applies only to Tasks with subtasks.

Roles and pools

A *role* is a local name for a type of worker (software agent or user) to which tasks are assigned. Every process has an *Owner* role that is responsible for the process as a whole.

One or more tasks are assigned to a role. The role is assigned to a worker or group of workers by associating the role with a *pool*. Each task is performed by a worker which is a member of the appropriate pool.

Every user must belong to at least one pool. Users can belong to several pools. In order to manage process instances, a user must be a member of the same pool as the eLink agent which started the process.

The eLink Business Process Option's default configuration makes this relatively transparent.

Process instance life cycle

The life cycle stages of a typical process instance are as follows:

1. **Process instance creation.** An eLink agent starts a new process based on a process definition.
2. **Task performance.** Tasks are performed in a specified order, and each task is performed by a worker agent who belongs to the pool that is assigned to the task.

3. **Process instance completion and archiving.** The process is completed when all tasks are finished, and completed process instances are recorded as necessary.

Process definitions

A process definition organizes tasks into a structure that determines when each task is to be performed. eLink directly supports two kinds of process definitions:

- Flat (default)
- Conditional

Flat definitions

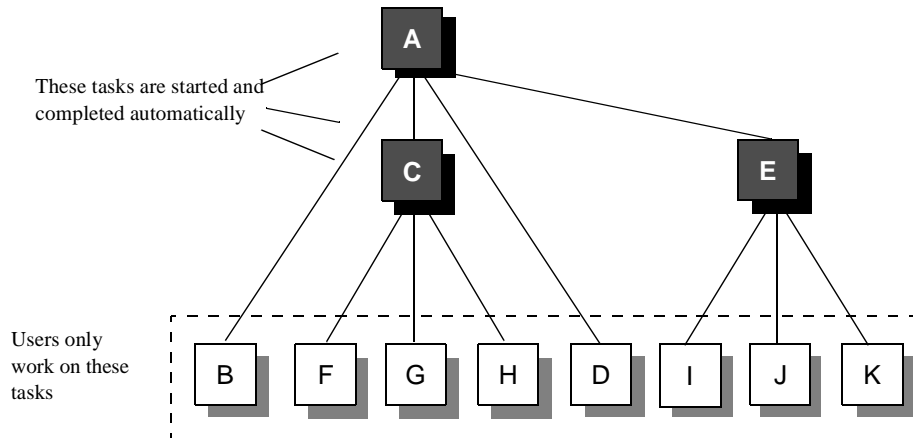
In a typical process definition, related tasks have a common parent. It can be convenient to use the common parent as a gateway task.

You can use a parent task as an organizing tool with no real work associated with it. This “flat” model of process organization is implemented through the task *AutoActivate* property. The model is called flat because effectively it creates a single-level task hierarchy.

Warning: By default, the Business Process Engine turns on the *AutoActivate* property for all tasks in a process. The eLink Business Process Option relies on this default behavior. You may also turn off the *AutoActivate* property for individual tasks, separately from the parent process. Tasks with subtasks must have *AutoActivate* on for the eLink Business Process Option to work properly.

In the flat model, tasks with subtasks start automatically when they become ready to do (they are in the *activated* state), and complete automatically when all their subtasks are complete.

Figure 2-1 Flat process definition



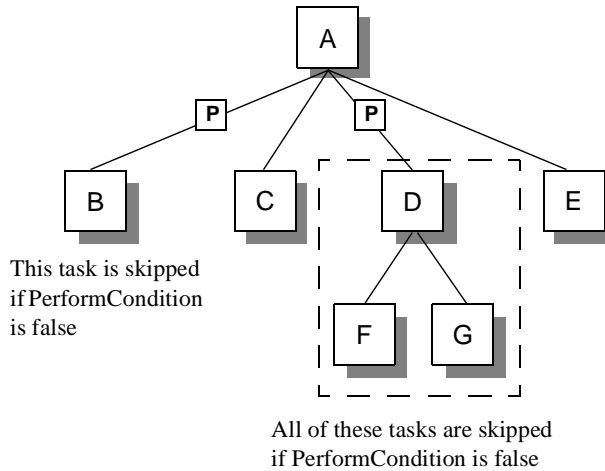
Conditional definitions

You may also create process definitions where tasks are performed only under certain circumstances — a so called *conditional workflow*. For example, it might be company policy to send a second billing notice to a customer when an invoice is more than 60 days overdue or process an insurance claim differently if the claim amount is over a million dollars.

Conditional workflows are implemented through *perform conditions*. A perform condition lets you specify a condition to be tested when a task becomes ready to do. The Business Process Engine can then decide whether to skip the task or let it be performed.

Figure 2-2 shows a process definition with two perform conditions (indicated by the letter “P” within the small boxes). If the perform condition on task B is false, that task is skipped. If the perform condition on task D is false, tasks D, F, and G is skipped. Thus, if a skipped task has subtasks, its subtasks are skipped, too.

Figure 2-2 Conditional process definition



You can also implement branching by defining opposite perform conditions on a pair of sibling tasks. For example, you might set up one task so that it is performed when the cost of an item is greater than a thousand dollars, and another task so that it is performed when the cost is less than or equal to a thousand dollars.

Perform conditions test the value of user-defined information called *attributes* (for more information, see “About attributes” on page 10).

A perform condition consists of an attribute name, an operator (greater than, equal to), and a value. The value can be another attribute. You can create complex perform conditions by combining tests with logical operators, for example:

```
CUSTOMER_NAME <> 'Kelly' AND COST > 500
```

```
MANAGER_APPROVAL = TRUE OR CLAIM_AMOUNT < 100
```

Table 2-2 shows examples of perform conditions.

Table 2-2 Example perform conditions

Attribute	Operator	Value
COST	>	1000

Table 2-2 Example perform conditions (Continued)

Attribute	Operator	Value
CLAIM_AMOUNT	<	CLAIM_MAX
CUSTOMER_NAME	=	'Kelly'
INVOICE_DUE_DATE	>	DATETIME(1995-10-21)
MANAGER_APPROVAL	<>	TRUE

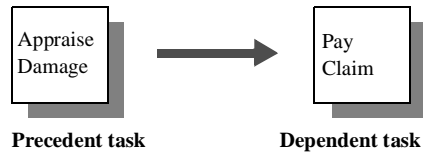
Chapter 5, “Making eLink Processes Work” the Working with Conditions section provides complete information on writing perform conditions.

Task dependencies

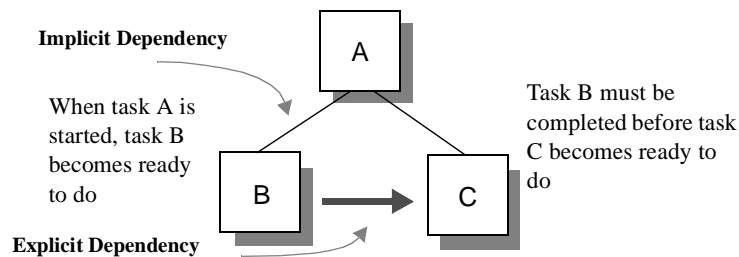
The process definitions in Figure 2-1, and Figure 2-2 are incomplete without further indication of the task sequence. In any process definition, subtasks become ready to do only after their parent has been started. (Recall that in a flat process definition, parent tasks start automatically.) This means that subtasks are *implicitly* dependent on their parent task. But what about sibling tasks?

Typically, some sibling tasks must be completed before others can be started. For example, an insurance company is unlikely to pay a claim before the damage has been appraised. In this case, payment is dependent on the appraisal.

In a task structure, you indicate such an *explicit* dependency by drawing an arrow from the precedent task to the dependent task, as shown in Figure on page 9. The Business Process Option supports explicit dependencies only between sibling tasks with the same parent.

Figure 2-3 Dependency between sibling tasks

When you start a process, The Business Process Engine does not let a dependent task become ready to do until its predecessor task is complete. The task that is parent to both sibling tasks need only be started, not finished, before its first subtask becomes ready to do. These relationships are illustrated in Figure 2-4.

Figure 2-4 Implicit and explicit dependencies

Kinds of task states

Each task in an active process has a *state*, which indicates whether its work can be performed at the present time. At any time, a task can be in one of these states:

Waiting

A task is waiting if another task must be performed first. No work can be performed on a waiting task. This is a task's initial state.

Ready

A worker can work on the task. However, no worker has yet started working on the task.

Bypassed *or* Skipped

The task's perform condition indicates that no work should be performed on the task, or the task was manually skipped.

Activated

As a result of a task's AutoActivate property, the task was started automatically and will be completed automatically when all its subtasks are completed.

In Process *or* Acquired

A worker is working on the task.

Done

The task is completed. No more work can be done on the task.

About attributes

Much of the information in the Business Process database is stored in the form of attributes. An *attribute* is information about a process or particular task within a process. Some attributes are built-in; that is, they are predefined for all systems. Others are user-defined. For example, if you were designing an insurance claims process, you might want to add information about the amount of the claim. You can add this information simply by creating a new attribute.

There is a third type of attribute--generated. These are the task attributes that are created (as part of the palette generation process) from operation parameters in the Business Interface Window within the eLink Process Design Assistant.

Built-in and user-defined attributes are used directly to store information during process execution. Generated attributes are used indirectly to tell the eLink Business Process Option agents how to process the information stored in built-in and user-defined attributes. These generated attributes contain parameter assignment expressions rather than business data.

For more information on parameter attributes and how they are used to implement process flows, see Business Process and Task Parameter Attributes in Chapter 5, "Making eLink Processes Work."

All attributes have two parts: a name and a value. When you create an attribute, you specify both its name and value. For built-in attributes, you specify only the value.

Table 2-3 shows examples of built-in attributes.

Table 2-3 Sample built-in attributes

Name	Type	Value
Task Name	string	Credit Check
Due Date	datetime	05/21/95
Priority	string	Normal
User Working On	string	Kelly

Table 2-4 shows examples of user-defined attributes.

Table 2-4 Sample user-defined attributes

Name	Type	Value
CLAIM_AMOUNT	integer	500
CUSTOMER_NAME	string	Kelly, K.
APPROVAL_CODE	string	22-A65

Table shows examples of generated attributes.

Table 2-5 Sample generated attributes

Name	Type	Value
AMOUNT	String	AMOUNT[*]=\$JOB.AMOUNT[*]
ACCOUNT_NUM	String	ACCOUNT_NUM[*]=\$JOB.ACCOUNT_NUM[*]
RESULT_BAL	String	\$JOB.BALANCE[*]=RESULT_BAL[*]

As these examples suggest, attributes can take several kinds of values: integer, string, datetime, and so forth. When you create a new attribute, you indicate what kind of value it can have.

Distinguishing process and task attributes

A *process attribute* applies to the process as a whole and, therefore, to all tasks that make up the process. When you set a process attribute, its value is the same in every task. Most user-defined attributes are process attributes. There are also built-in process attributes. For example, the process name is a built-in process attribute that you set when you create a process.

A few built-in process attributes are read-only for all users. This means that only the Business Process Option can assign the attribute value. For example, when you start a process, the Time Created attribute is automatically set to the current time.

Task attributes are unique for each task. Many built-in task attributes are provided; some of these are listed in Table 2-3. You cannot create new task attributes during process design.

You can set some task attributes during process design. Others are read-only. For example, you can set the task Due Date attribute when you design (or start) a process. But only The Business Process Engine can set the Time Started attribute when a worker begins to work on a task.

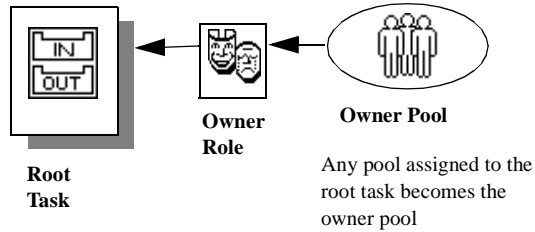
About process ownership

Every active process must have an owner. The process owner typically is responsible for managing the process and monitoring its activity. eLink agents are the owners of all processes within the Business Process Option.

Therefore, to manage a process manually you must become a member of the *owner pool*. The owner pool is the pool you assign to the root task.

As shown in Figure 2-5, any pool can be the owner pool. If you do not specify an owner pool in the process definition, it defaults to the process initiator's pool.

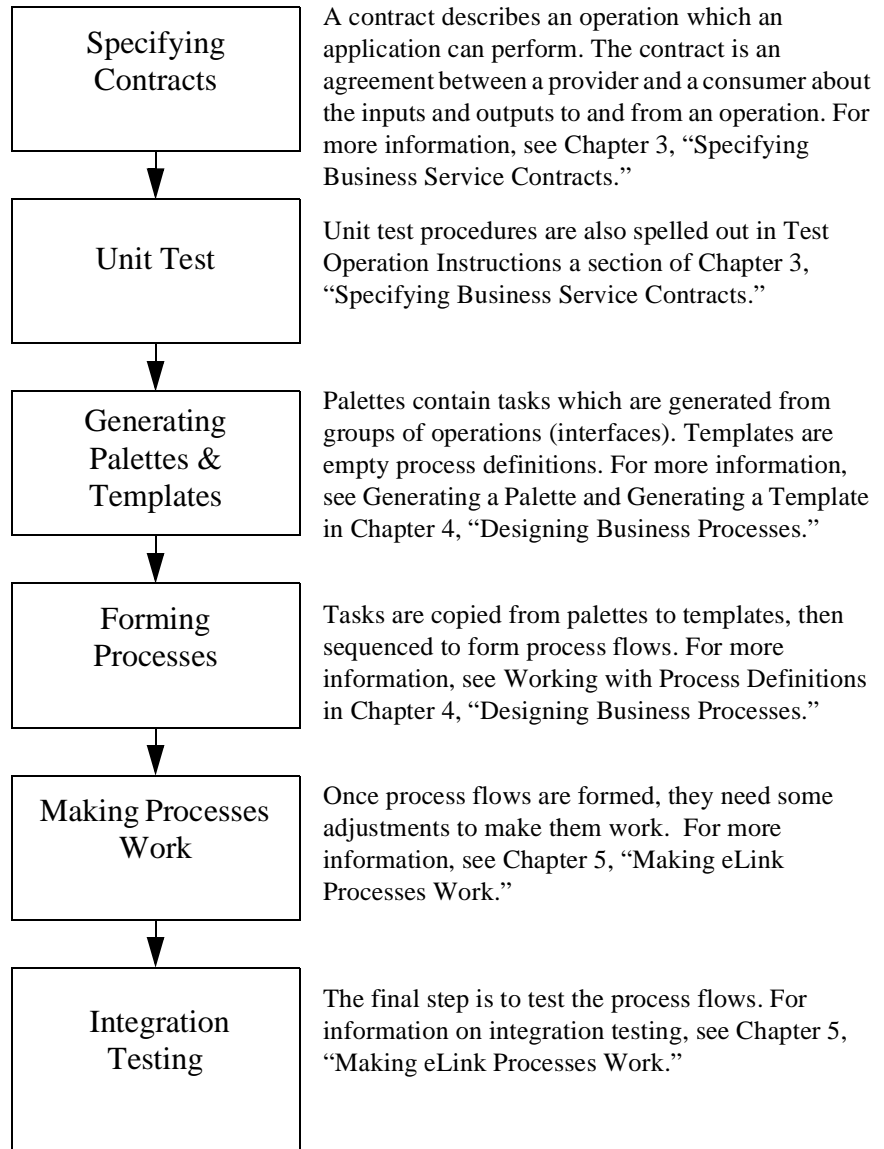
Figure 2-5 Owner role and owner pool



Creating Business Processes: An Overview

The following flow chart is an overview that describes the step-by-step procedure for forming an eLink process. This is the only correct procedure for forming an eLink process.

The following chart outlines the steps necessary for creating business processes using the eLink Business Process Option:



3 Specifying Business Service Contracts

The following topics are covered in this chapter:

- Overview
- Business Interface Window
- Usage
- Process Design Assistant and ATMI Mappings

Overview

The Process Design Assistant is a tool that assists you in creating contracts between applications, then forming them into process flows. This chapter provides instructions for the Business Interface Window of the Process Design Assistant. The Business Interface Window assists you in specifying contracts.

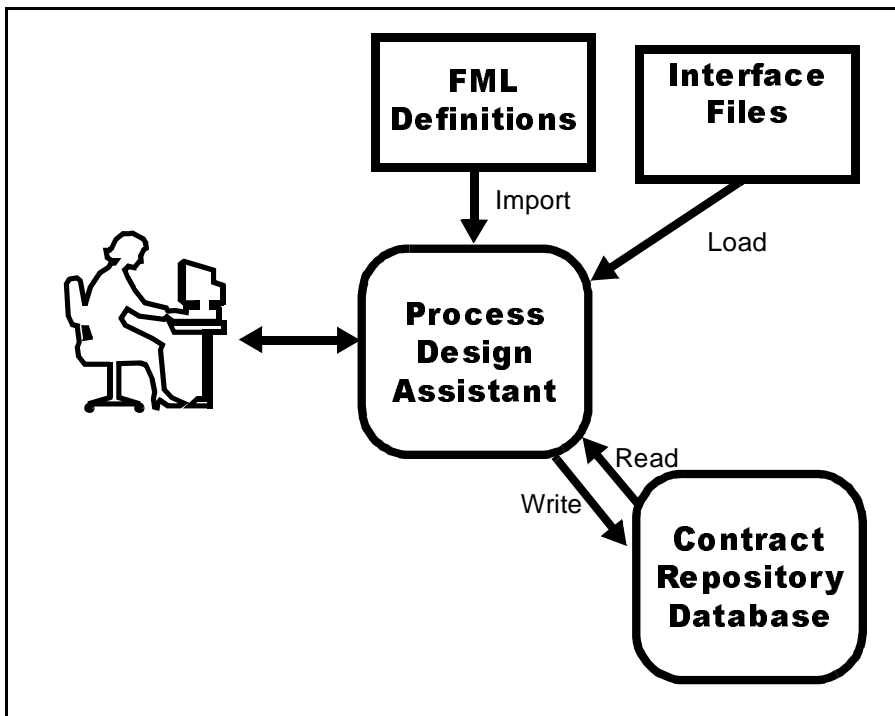
How It Works

The Process Design Assistant is an eLink client/server application. Contracts are specified in the Business Interface Window, then formed into process flows in the Business Process Window. You can also import existing FML definitions (to be used as the basis for contracts) and load interfaces from text files. See Appendix A, “Interface File (*.IFCE) Reference,” for more information.

The Process Design Assistant supports the description of any eLink service that uses FML32 buffer types.

Figure 3-1 shows the workflow using the Contract Repository database. The Process Design Assistant (Business Interface Window) and other key components are described in the next section.

Figure 3-1 Workflow for Specifying Contracts



Key Components

The Process Design Assistant consists of the following components:

- **Contract Repository Database**—The Contract Repository database stores interface contracts between clients and servers. All created objects reside in the database and are used to design processes.
- **The Business Interface Window**—The Business Interface Window of the *Process Design Assistant* is the primary interface to the Contract Repository database. The contents of the database are shown in an object tree structure that shows the available modules, the interfaces contained in each module, and so on. You can use the Interface Window to add, modify, or delete the objects in the database. You also have the option to populate the Contract Repository database by importing existing FML files or load interfaces from text files. See Appendix A, “Interface File (*.IFCE) Reference,” for more information on loading interfaces from text files.

To learn more about the objects in the Contract Repository tree, refer to the section of this chapter on the Contract Repository Tree View.

For more information about working with objects in the repository, refer to the section of this chapter on Usage.

- **The Business Process Window**—The Business Process Window is used to form process flows. Process flows are formed using palettes and templates in the Design Pad (subwindow). The process flows formed are displayed as flow diagrams in the Design Pad. The process boxes in the diagrams represent tasks and the arrows in the diagrams represent dependencies between tasks.

Business Interface Window

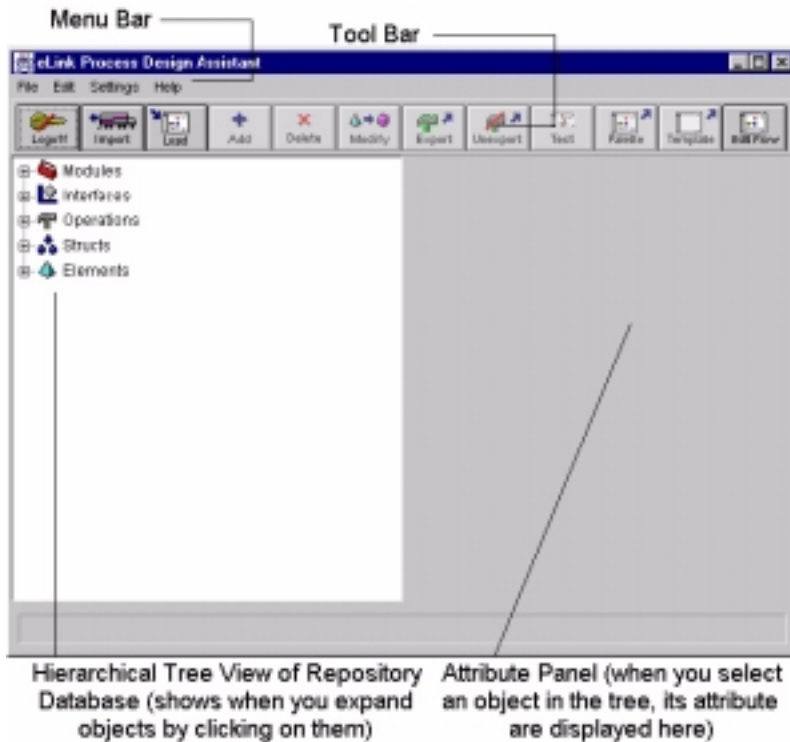
The Business Interface Window shows a tree view of the contracts and their components. When you select an item in the tree, its properties are displayed in the right-hand panel.

Before you can view the tree in the Process Design Assistant, you need to log on. To do this, select **File**—>**Logon** from the menu bar or click on the Logon toolbar button.

Accessing Contract Repository Objects

The Process Design Assistant Business Interface Window consists of the menu bar, toolbar and an attribute display. After logon, the root nodes for each Contract Repository object are shown. Figure 3-2 shows the root nodes for each type of object in the Contract Repository tree that display in the left portion of the window.

Figure 3-2 Main Tree Structure



Contract Repository Tree View

As shown in Figure 3-2, there are multiple roots in the tree, one root for each kind of object.

- The first root consists of modules. Modules are optional grouping structures. They will help you organize interfaces but are not needed for process creation. Interfaces provide the only grouping necessary to generate palettes.
- The second root displays the interfaces defined in the Contract Repository. Each interface may be part of one or more modules. An interface groups together one or more related operations (for example, all of the business services offered by an accounting application). Once created, an interface can be used to generate a palette of tasks.
- The third root is operations. An operation may be part of one or more interfaces. Once created, an operation can be used to generate a template to form process flows.
- The fourth root displays the structs defined in the repository. The structs are groups of elements. A struct can be used as a data type for elements.
- The fifth root is elements. The element type can be a basic type (char, short, long, float, double, string, array) or a struct type. Parameters in operations are defined using elements.

From the tree view, you can create elements, structs, operations, and interfaces.

As an example, Figure 3-3 shows the BANKAPP module expanded. Table 3-1 describes each type of object available in the tree in full detail.

Figure 3-3 Module Tree Hierarchy

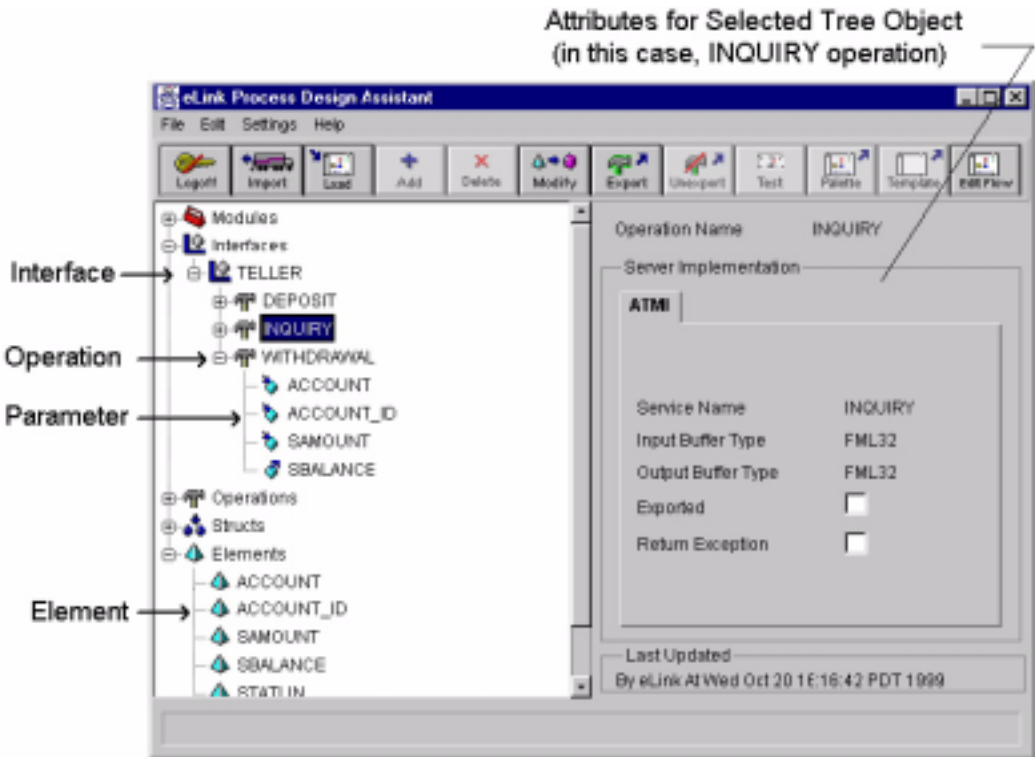


Table 3-1 Tree Objects




Object	Icon	Description
Module	 module	A module is a group of one or more interfaces. In eLink, this object type is used only to group and organize interfaces.
	 out of date module	
Interface		Interfaces are a group of one or more operations. Each interface may be part of one or more modules.

Table 3-1 Tree Objects














Object	Icon	Description
Operation	 operation	Operations are implemented as ATMI services. Each operation may be part of one or more interfaces.
	 exported operation	When an operation has been <i>exported</i> it shows in a green light.
Parameter		When elements are used in operations, they become <i>parameters</i> or arguments to the operation. Parameters contain additional information (beyond that in the corresponding element), such as passing mode, exception mode, etc.
		A parameter is based on a single primary data type (e.g., short, float, char) or a struct.
		The parameter attributes available are:
	 input	<ul style="list-style-type: none"> ■ <i>input</i>, <i>output</i>, and <i>input/output</i> — <i>input</i> sends a value to the operation; <i>output</i> and <i>input/output</i> return a value to the calling operation
	 output	<ul style="list-style-type: none"> ■ <i>no access</i> — <i>no access</i> neither sends a value or returns a value to the operation
	 input/output	<ul style="list-style-type: none"> ■ <i>output returns only on exception</i>— special case of <i>output</i> which returns a value only if there is an exception caught
	 no access	<ul style="list-style-type: none"> ■ <i>output always returns including on exception</i>— special case of <i>output</i> which always returns a value including when an exception is caught
	 output returns only on exception	<ul style="list-style-type: none"> ■ <i>input/output always returns including on exception</i>— special case of <i>input/output</i> which always returns a value including when an exception is caught
	 output always returns including on exception	<ul style="list-style-type: none"> ■ <i>input/output returns only on exception</i>— special case of <i>input/output</i> which only returns a value including when an exception is caught
 input/output always returns including on exception	<ul style="list-style-type: none"> ■ <i>optional exception parameter</i>— optional exception parameter that can be used for error checking 	
 input/output returns only on exception		
 optional exception parameter		

Table 3-1 Tree Objects

Object	Icon	Description
Struct		A struct is a group of previously defined elements. To use a struct as a parameter, define it as an element. Structs cannot be nested.
Element		An element is based on a single primary data type (e.g., short, float, char) or a struct. An element can be used by itself as a parameter to one or more operations, as well as being a part of one or more struct types.

Refreshing the Contract Repository Tree View

During development time, the Contract Repository database may be frequently updated as objects are added, imported, deleted, or modified (by you or other eLink Process Design Assistant users). You must *refresh* the Process Design Assistant in order to see any changes made after you log on (that is, during a session).

To refresh the Contract Repository, choose **File**—>**Refresh** from the menu bar. This updates your view of the Contract Repository database. Note that if the object tree was expanded before you refreshed, it will re-display showing only the initial root nodes. When you re-expand the tree, you will see the updated version.

Toolbar Buttons

The Process Design Assistant Business Interface Window displays a series of icons, or toolbar buttons, under the menu bar used to quickly access frequently used options. Table 3-2 describes the toolbar buttons.

Table 3-2 Toolbar Buttons














Button	Label	Function
	Logon	Logs on to the Process Design Assistant Business Interface Window.
	Logoff	Logs off the Process Design Assistant Business Interface Window.
	Import FML	Imports FML files.
	Load	Loads previously defined operations.
	Add	Adds a module, interface, operation, struct, or element.
	Delete	Deletes a module, interface, operation, struct, or element.
	Modify	Modifies a module, interface, operation, parameter, struct, or element.
	Export	Makes the operation available for testing and other purposes.

Table 3-2 Toolbar Buttons

Button	Label	Function
	Unexport	Makes the operation unavailable for testing and other purposes.
	Test	Tests the operation.
	Palette	Generates a Palette.
	Template	Generates a Template.
	Edit Flow	Opens the Business Process Window. The Business Process Window is where you can design and edit a business process flow.

Keyboard & Menu Shortcuts

The following table shows the keyboard and menu shortcuts available on the Process Design Assistant.

Action	Keyboard Shortcut	Menu or Tab Path
Log on to the Process Design Assistant Business Interface Window.	Ctrl L	File—>Logon

Action	Keyboard Shortcut	Menu or Tab Path
Import an FML file into the Contract Repository database.	Ctrl I	File—>Import FML
Loads an interface from a text file called an interface file. For more information on interface files, see Appendix A, “Interface File (*.IFCE) Reference.”		File—>Load
Refresh the Process Design Assistant object tree from the Contract Repository database.	Ctrl R	File—>Refresh
Log off the Process Design Assistant Business Interface Window.	Ctrl L	File—>Logoff
Exit the Process Design Assistant Business Interface Window.	None	File—>Exit
Add a module, interface, element, or structure to the selected root tree of kind of object you want to add.	Ctrl Insert	Edit—>Add
Delete the selected object.	Ctrl Delete	Edit—>Delete
Modify the selected object.	Ctrl M	Edit—>Modify
Export the selected object.	Ctrl E	Edit—>Export
Unexport the selected object.	Ctrl E	Edit—>Unexport
Test the selected operation for run-time functionality.	Ctrl T	Edit—>Test
Generates Palette from the selected interface.		Edit—>Generate Palette
Generates Template from the selected operation.		Edit—>Generate Template

Action	Keyboard Shortcut	Menu or Tab Path
Opens the Business Process Window which allows you to edit process flows.		Edit—>Edit Flow
Set default logon values.	Ctrl U	Settings—>Logon
Bring up the online help	F1	Help—>Online Help

Pop-up Menus for Tree Objects. In addition to the menus and toolbar buttons, you can get a pop-up menu of actions for a selected tree object. To do this, select an item in the tree on the Business Interface Window and click the right mouse button to display the actions. Highlight an action on the menu and click the right mouse button again to select it.

Usage

The Process Design Assistant is a tool that assists eLink users in specifying business processes.

This section explains how to work with the Repository Database using the Process Design Assistant.

The following topics are covered:

- **Starting the Process Design Assistant.** Provides instructions for starting the Process Design Assistant Business Interface Window.
- **Logging On to the Process Design Assistant.** Provides instructions for customizing your logon default settings, then logging on to the Process Design Assistant.
- **Importing FML Files.** Describes the FML file import as a batch-load process that imports existing FML files into the Contract Repository. During the load process, duplicate entries are updated in the Contract Repository using the new information provided in the import file. If you plan to load externally described

interfaces, you must import the FML files that contain the definitions for the FML fields used by the interface.

- **Loading Externally Described Interfaces.** Describes how you can load an interface from a text file called an interface file. For more information on interface files, see Appendix A, “Interface File (*.IFCE) Reference.” If you plan to load externally described interfaces, you must import the FML files that contain the definitions for the FML fields used by the interface.
- **Creating New Objects.** You can create modules, interfaces, operations, parameters, structs, and elements using the Process Design Assistant.
- **Modifying Objects.** You can modify modules, interfaces, operations, parameters, structs, and elements using the Process Design Assistant.
- **Deleting Objects From the Repository.** You can delete modules, interfaces, operations, structs, and elements using the Process Design Assistant. You cannot delete objects that are referenced by other objects in the Contract Repository database.
- **Exporting or Unexporting an Operation.** You can export or unexport operations. Export serves as a flag that makes the object available for testing, generating palettes and templates, and web apps for later use.
- **Testing an Operation.** Describes how you can test an operation and its parameters to ensure that all components are functioning properly before you can use it further.
- **Logging Off and Exiting** the Process Design Assistant Business Interface Window.
- **Troubleshooting.** You are notified of any error that occurs by an Alert message. The message prompts you for an action or advises you of the error.

Logging on and Getting Started

Before starting the Process Design Assistant, make sure that you have installed all necessary software. For details on system requirements, supported platforms, and Process Design Assistant software installation, refer to the *BEA eLink Business Process Option Operations & Maintenance Guide* included with your CD-ROM.

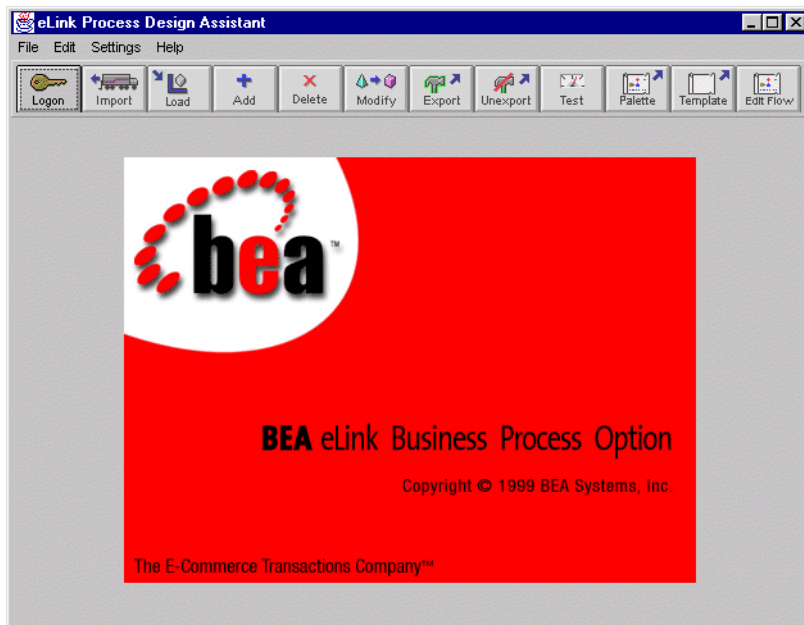
Starting the Process Design Assistant

To start the Process Design Assistant:

1. Select the **BEA eLink** folder icon from your directory tree.
2. Select the **eLink Process Design Assistant** icon from the BEA eLink folder. The opening window in Figure 3-4 displays. (If you would like to create a Windows shortcut, see Help for your version of Windows.)

The BEA eLink Business Process Option opening window (shown in Figure 3-4) appears.

Figure 3-4 Process Design Assistant at Start Up



Logging On to the Process Design Assistant

Before logging on for the first time, you must set certain logon defaults as this information is not provided by the logon dialog. If you don't set these options, you will be unable to log on to the Process Design Assistant.

Setting Logon Option Defaults

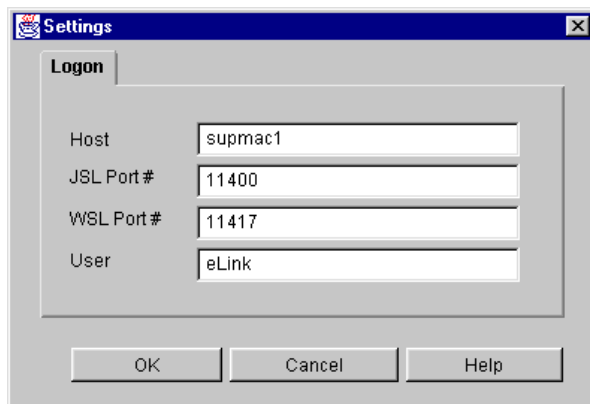
Warning: If the JSL and WSL Port numbers are reversed in error, the server may lock up so check with your System Administrator to make sure that you have the correct port numbers to enter in the Logon Settings Window.

You must set your logon default settings before logging on to the Process Design Assistant.

Note: Setting the logon option defaults need only be done when a new eLink Business Process Option user logs on to the Process Design Assistant for the first time, or should the host access information change.

The first three fields in the Logon settings window must be set up to provide host information for logons. Optionally, the user name may be set as well.

Figure 3-5 Logon Settings Window



Setting the Logon Default Instructions

To set the logon default:

1. Choose **Settings** —> **Logon** from the menu bar.
2. The Settings window in Figure 3-5 displays.
3. Type the values for **Host**, **JSL Port #**, **WSL Port #**, and **User** (30-character limit). The **Port #** fields must be integers. If you do not know this information, ask your System Administrator.
4. To set the values, click **OK**. Click **Cancel** to return to the previous window.

Logging On

Logon Instructions

After starting the Process Design Assistant and setting the logon defaults (required), follow the directions to log on:

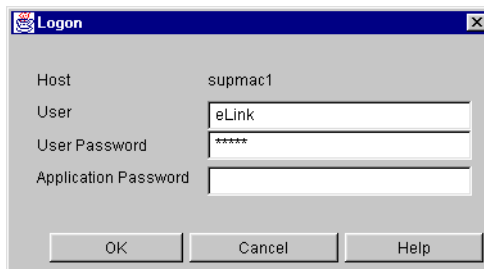
1. Choose **File** —> **Logon** from the menu bar

Or

Click on the Logon toolbar button.

2. The Logon window shown in Figure 3-6 displays with text entry fields. Complete these fields to establish a connection with the machine offering the Contract Repository services.

Figure 3-6 Logon Window



3. Type the password in the **Password** field, if required by your server configuration.
4. Type the application password (8-character limit) in the **Application Password** field if required by your server configuration.
5. Click **OK**.

Note: The access information for the host displayed in the Logon window must be set up as defaults in the Settings window. See *Setting the Logon Default Instructions*.

Importing FML Files

The FML file import is a process that imports existing FML files into the Contract Repository. These FML files are maintained by your system administrator.

Note: FML files describing the FML fields that you will be using in eLink (as elements) must be imported before those fields can be used in operations. However, you should import an FML file describing particular FML field definitions only once to avoid overwriting subsequent changes to the operational parameters (elements).

You can import additional new files into the Contract Repository on an ongoing basis as the need for new interfaces arises. If you are unsure as to whether or not a particular FML field definition has been imported, check the properties of the corresponding element by highlighting it in the repository tree. Its properties will appear on the right side of the Business Interface Window. If an element has a non-zero FML field number, its field definition has been imported.

Note: For information on the creation of the FML files, refer to the *BEA TUXEDO Programmer's Guide*.

When you import FML fields, they are imported as elements. During the import, duplicate entries are updated in the Contract Repository using the new information provided in the import file. The **Occurrence** default is "1."

To import FML files, use the Import FML file window (shown in Figure 3-8.) The window has two display areas: **Importable Files** and **Files for Import**. The **Importable Files** display area contains existing FML files that currently reside on your server.

You can select one or more of the files by using the arrow keys to move the files between the display areas. If you move a file to the **Files for Import** display area, and then decide you do not want to import it, use the arrow keys to move it back to the **Importable Files** display listing.

Figure 3-7 shows how the import process works. The FML files are imported into the Contract Repository database. If you want to see the objects that you have just imported, you must refresh the tree structure of your GUI by selecting **File** → **Refresh** from the menu bar after the import.

Figure 3-7 FML Import Procedure

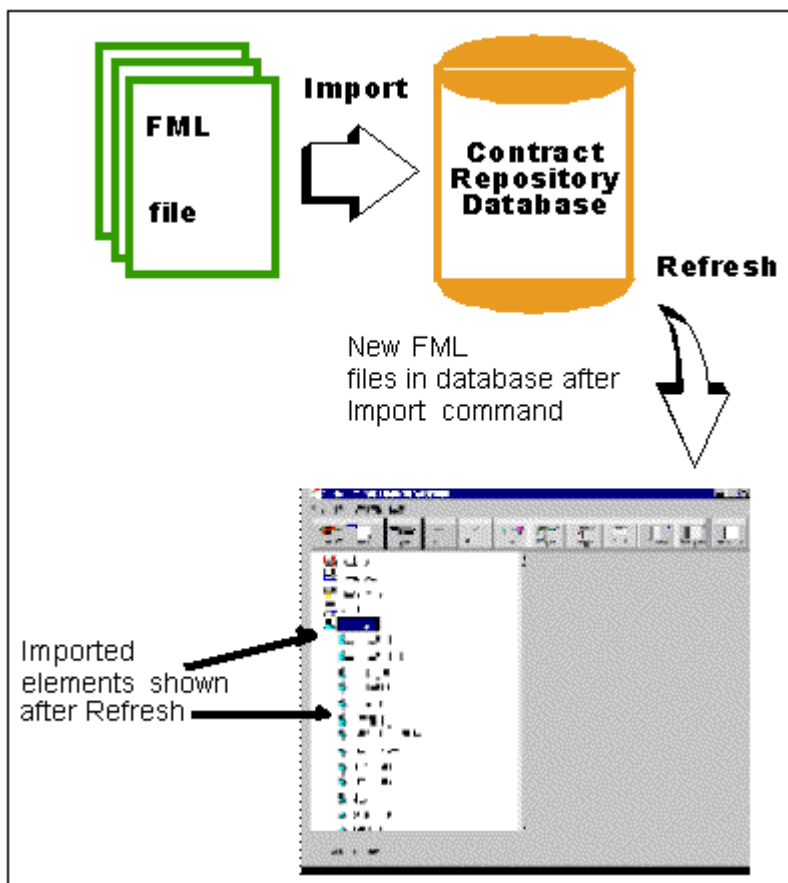
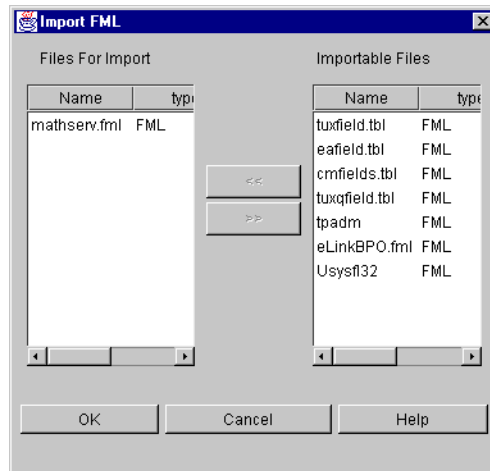


Figure 3-8 FML File Import Window



FML Import Instructions

To use the FML import:

1. Choose **File** —> **Import FML32** from the menu bar

Or

Click on the Import toolbar button.

2. A window listing the importable FML files displays.
3. Select the files to import. Use the << or >> buttons to move the files between the display areas.

Click **OK** to begin the import process. You can track the status of the import by referring to the messages displayed in the lower left-hand corner of the interface. (Or click **Cancel** if you decide not to import at this time.)

4. If the import has not started and you wish to discontinue this process, press **Cancel** to return to the previous window.

Note: You cannot cancel an import once it has started. If you inadvertently start the import, wait until it has completed the import cycle. If the import stops due to an error, you are notified by the Alert dialog box. For more

information on the Alert window, refer to the Troubleshooting section in this chapter.

5. When the import is finished the message in the dialog box prompts you to answer, “Do you want to refresh now?” To refresh, click **OK**.

If you want to check for any problems encountered during the import, select **View Log** to view the import activity log before refreshing the GUI.

If you imported a file that contained elements that you do not need, you can delete them. Refer to *Deleting Objects From the Repository* section in this chapter for information regarding object dependencies. Objects that are dependent on other objects cannot be deleted.

Loading Externally Described Interfaces

As an alternative to interactively defining operations, then grouping them into interfaces, you may wish to load text files describing these operations.

Instructions for Loading Externally Described Interfaces

To load interfaces:

1. Choose **File** —> **Load Interface** from the menu bar.

Or

Click on the Load toolbar button.

2. The Load Interface file selection window displays.
3. Select the interface file that you want to load into the Process Design Assistant, then click **OK**.

A log box will appear indicating the load actions.

4. When the load completes click to check the Refresh Repository checkbox, then click the **OK** button.

The newly loaded interfaces will appear in the Contract Repository tree on the left side of the Business Interface Window.

Creating New Objects

You can create new objects by using the Add option from the Edit menu.

When you populate the Contract Repository, we suggest that you construct your objects from the “bottom up.” Create your elements first, then structs, operations, interfaces, and finally, modules. Elements are self-contained, but each of the other objects depend on the object prior to it in the Contract Repository tree (e.g., structs depend on elements).

Follow the instructions in the sections for:

- **Format Rules for Repository Objects**
- **Adding Elements**
- **Adding Structs**
- **Adding Operations**
- **Adding Interfaces**
- **Adding Modules**

Format Rules for Repository Objects

All object names that you input must conform to the format rules of C language identifiers. Each identifier must start with an alphabetic character or an underscore (e.g., `fixed_occurrence`). Each identifier can include alphabetic characters, underscores, and numeric characters.

Adding Elements

You create new elements by using the Add Element window shown in Figure 3-9.

Figure 3-9 Add Element Window

Table 3-3 Add Element

Field	Description
Element Name	Identifies the name of the new element. You must enter the name of the element you wish to create. This field cannot be left blank or contain spaces. The name must not exceed 32 characters.
Element Type	Indicates the element type; either a primary element type (char, string, short, long, float, double, carry) or a struct type. If an element is a member of a struct, you cannot select an element type that is a struct. To select an element type, click on the Select button next to the Element Type field. (For more information, see Selecting an Element Type.)
Element Length	Enabled when an element type is string or carry. You cannot edit this field for other types.

3 Specifying Business Service Contracts

Table 3-3 Add Element

Field	Description
Occurrence	The number of occurrences of this element. If 0, this indicates an unlimited number of occurrences for the element. If the element is an instance of a struct type, you must indicate "1" for the occurrence and it cannot be edited. The occurrence must not exceed 9 characters. The occurrence number determines the number of data entry fields in the Test Operations dialog box. This is the only effect of this attribute.
Fixed	If the element type is struct type, you must indicate its occurrence as "1," and it must be fixed (if you selected the checkbox). For all other element types, if the occurrence is "0," it must not be fixed (if you did not select the checkbox). If the occurrence is "1," it must be fixed.
Comments	Allows entry of one line of comment text, without line breaks. You cannot include a colon (:) in the comments.
FML Field Name	Shows the FML field name. To select the FML field name, click on the Select button next to the field. For more information, see Selecting an FML Field Name.
FML Field Number	Shows the FML field number for this FML field.
Last Updated	Displays the date and time of the most recent update and the name of the person who made the update. This area is empty until the element is created.
OK	Saves your addition.
Cancel	Returns to the previous window.
Help	Accesses online help.

Adding Elements Instructions

To add an element:

1. Select **Elements** root in the main directory tree.
2. Choose **Edit** —> **Add** from the menu bar
Or
Click on the Add toolbar button.
3. The Add Element window displays.

4. Complete the text entry fields and click **OK** to save the new element. Press **Cancel** to return to the previous window.

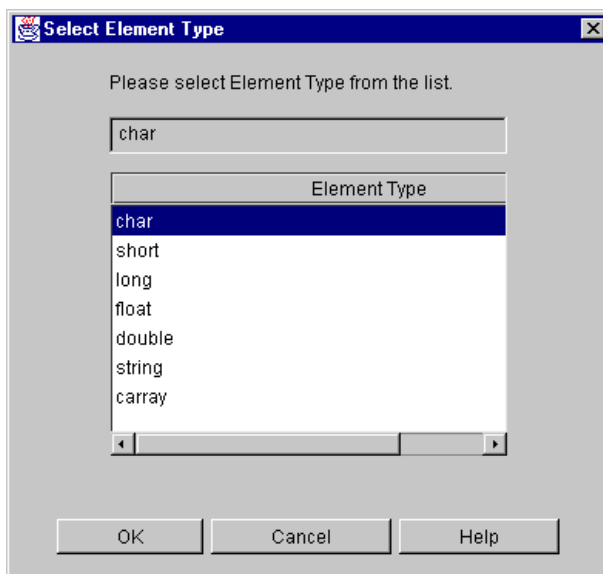
Selecting an Element Type

To select an element type from the Adding Elements window:

1. On the Add Element or Modify Element window, click **Select** next to the Element Type field.

This brings up the Select Element Type window as shown in Figure 3-10.

Figure 3-10 Select Element Type Window



2. Use the scroll bar to view the available element types.
3. Select an element type and click **OK** to save your choice and return to the Add Element or Modify Element window. (Or click **Cancel** to close the window without saving your changes.)

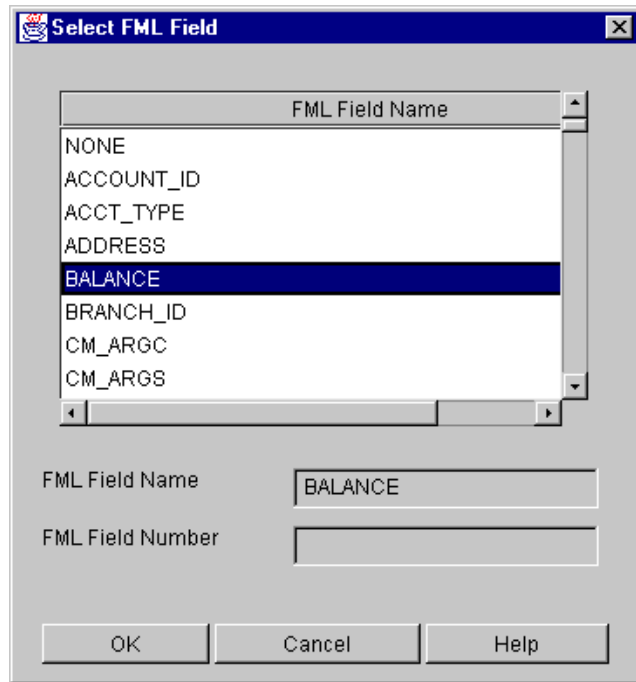
Selecting an FML Field Name

To select an FML field name:

1. On the Add Element or Modify Element window, click **Select** next to the FML Field Name field.

This brings up the Select FML Field window as shown in Figure 3-11.

Figure 3-11 Select FML Field Window



2. Review the list of available FML Field Names and select a field name from the list.
3. Click **OK** to save your changes. (Or click **Cancel** to close the window without saving your changes.)

Adding Structs

Structs (also referred to as “structures”) comprise a group of available element members. A struct cannot be nested within a struct. Each member has a name, type, and length information. Use the << or >> keys to move members between current and

available status. When a member name is moved, all of the information accompanying the member is moved. See Format Rules for Repository Objects for information about object naming conventions.

Figure 3-12 Add Struct Window

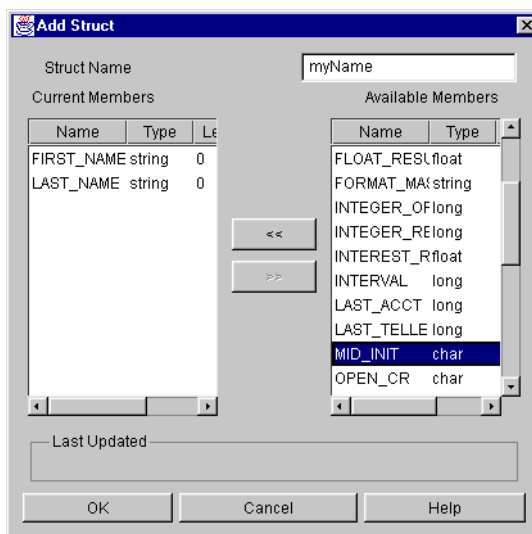


Table 3-4 Add Struct

Field	Description
Struct Name	Identifies the name of the struct. You must enter the name of the struct you wish to create that is unique and does not exceed more than 31 characters.
Current Members	Lists the current member names for the struct. The display area is blank for all new structs.
Available Members	Lists available member names for the struct.
Arrow keys (<< or >>)	Moves selected item between Current Members and Available Members .
Last Updated	Displays the date and time of the most recent update and the name of the person who made the update. This area is empty until the struct is created.
OK	Saves your addition.

Table 3-4 Add Struct

Field	Description
Cancel	Returns to the previous window.
Help	Accesses online help.

Adding Structs Instructions

To add a struct:

1. Select **Structs** in the main directory tree.
2. Choose **Edit** → **Add** from the menu bar
Or
Click on the Add toolbar button.
3. The Add Struct window in Figure 3-12 displays.
4. Complete the text entry fields and click **OK** to save the new struct. Press **Cancel** to return to the previous window.

Adding Operations

To add operations, the list of available parameters (based on previously defined elements) is displayed on the right scrollable list. Type the operation name in the text entry field, select the parameters, and enter the ATMI server implementation information. See Format Rules for Repository Objects for information about object naming conventions.

The **Edit** button is used to change a parameter's passing mode and mandatory attributes after it has been moved from the available list. See Modifying Parameters for additional information. The default fields are shown below.

Field	Default
Passing Mode	in
Mandatory	Yes

Field	Default
Display Name	Parameter name

Figure 3-13 Add Operation Window

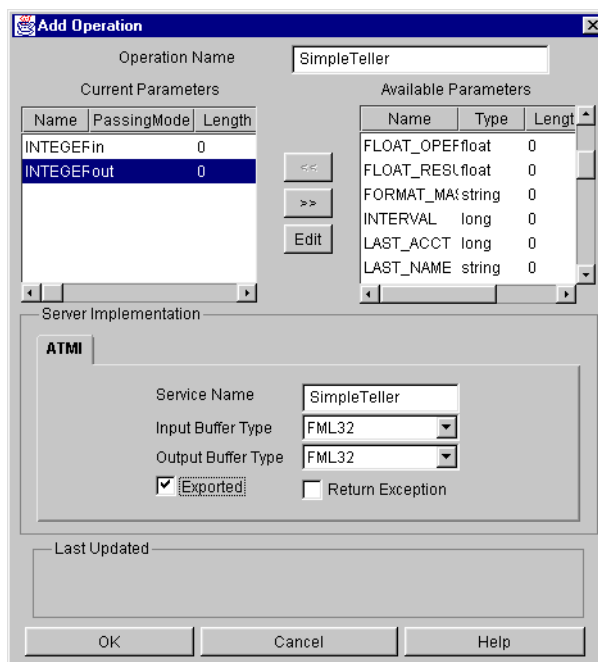


Table 3-5 Add Operation

Field	Description
Operation Name	Identifies the name of the operation. You must enter the name of the operation you wish to create. This field cannot be left blank or contain spaces. The name must not exceed 32 characters.

3 Specifying Business Service Contracts

Table 3-5 Add Operation

Field	Description
Current Parameters	<p>Lists the current parameter names for the operation.</p> <p>If the Current Parameters list contains a parameter that is a struct type, it is not allowed to have another parameter that is a member of the struct type parameter. In addition, if the Current Parameters list contains a basic parameter, it is not allowed to have another struct type parameter that contains a member of the basic parameter.</p>
Available Parameters	<p>Lists available parameter names for the operation.</p>
Arrow keys (<< or >>)	<p>Moves selected item between Current Parameters and Available Parameters.</p>
Edit	<p>Enables editing of a selected current parameter. This option is disabled if you have not selected a current parameter.</p> <p>When you select a “current parameter” and click Edit, you get the Edit Parameter window, which functions exactly the same as the Modify Parameter window.</p> <p>For information about how to edit the parameter, see Modifying Parameters.</p>
ATMI Service Name	<p>Identifies the ATMI service that implements the operation. It is the name used within the eLink environment to connect to business services either offered by or called by eLink adapters. The ATMI Service Name must not be left blank and must obey the ATMI service naming conventions.</p>
Input Buffer Type and Output Buffer Type	<p>FML32 — a type in which each field carries its own definition.</p> <p>NONE — placeholder indicating that no input or output is required.</p>
Exported	<p>Exported operations are made available for testing, generating palettes and templates, and web apps for use in run time. Select the checkbox to export the operation. If you do not select the checkbox, the operation remains unexported.</p>
Return Exception	<p>Enabled if exceptions are returned.</p>
Last Updated	<p>Displays the date and time of the most recent update and the name of the person who made the update. This area is empty until the operation is created.</p>
OK	<p>Saves addition.</p>
Cancel	<p>Returns to the previous window.</p>
Help	<p>Accesses online help.</p>

Adding Operations Instructions

To add an operation:

1. Select **Operation** in the main directory tree.
2. Choose **Edit** —> **Add** from the menu bar
Or
Click on the Add toolbar button.
3. The Add Operation window in Figure 3-13 displays.
4. Complete the text entry fields and click **OK** to save the new operation. Press **Cancel** to return to the previous window.

Adding Interfaces

You can create an interface by accessing the Add Interface window. The available operations are displayed on the right and the current operations are displayed on the left. Use the arrow buttons to move operations in or out of the interfaces. See Format Rules for Repository Objects for information about object naming conventions.

Figure 3-14 Add Interface Window

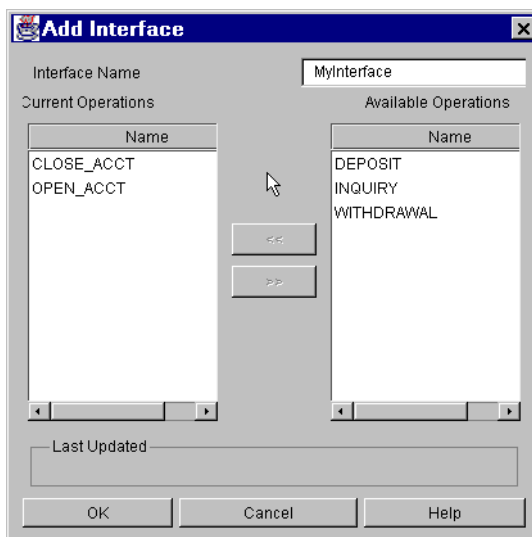


Table 3-6 Add Interface

Field	Description
Interface Name	Identifies the name of the interface. You must enter the name of the interface you wish to create. This field cannot be left blank or contain spaces. The name must not exceed 32 characters.
Current Operations	Lists the current operation names for the interface. The display area is blank for all new operations. The Current Operations list can remain empty.
Available Operations	Lists available operation names for the interface.
Arrow keys (<< or >>)	Moves selected item between Current Operations and Available Operations .
Last Updated	Displays the date and time of the most recent update and the name of the person who made the update. This area is empty until the interface is created.
OK	Saves your addition.
Cancel	Returns to the previous window.
Help	Accesses online help.

Adding Interface Instructions

To add an interface:

1. Select the root of the interface tree.
2. Choose **Edit** —>**Add** from the menu bar
Or
Click on the Add toolbar button.
3. The Add Interface window in Figure 3-14 displays.
4. Complete the text entry fields and click **OK** to save the new interface. Press **Cancel** to return to the previous window.

Adding Modules

Figure 3-15 shows the Add Module window. To create a module, select the root of the module tree, choose **Edit**—>**Add** from the menu bar to display the Add Module dialog box, and type the name in the **Module Name** text entry field. Select the module's current interfaces by moving items from the **Available Interfaces** display area to the **Current Interfaces** display area. See Format Rules for Repository Objects for information about object naming conventions.

Figure 3-15 Add Module Window

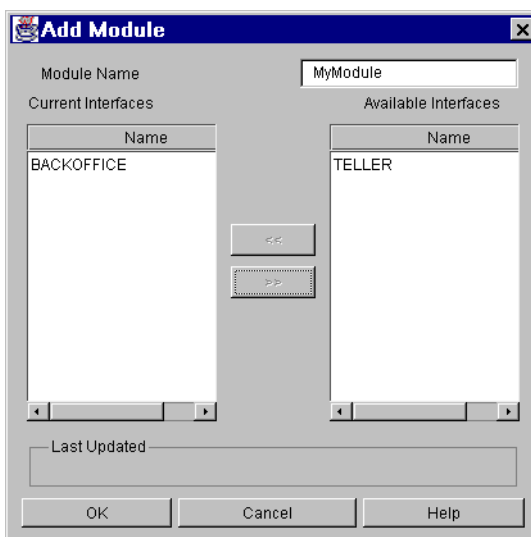


Table 3-7 Add Module

Field	Description
Module Name	Identifies the name of the module. You must enter the name of the module you wish to create. This field cannot be left blank or contain spaces. The name must not exceed 32 characters.
Current Interfaces	Lists the current interface names for the module. The display area is blank for all new modules. The Current Interfaces list can remain empty.
Available Interfaces	Lists available interface names for the module.

Table 3-7 Add Module

Field	Description
Arrow keys (<< or >>)	Moves selected item between Current Interfaces and Available Interfaces .
Last Updated	Displays the date and time of the most recent update and the name of the person who made the update. This area is empty until you create a module.
OK	Saves your addition.
Cancel	Returns to the previous window.
Help	Accesses online help.

Adding Modules Instructions

To add a module:

1. Select the root of the **Module** tree in the main directory tree.
2. Choose **Edit** —>**Add** from the menu bar
Or
Click on the Add toolbar button.
3. The Add Modules window in Figure 3-15 displays.
4. Complete the text entry fields and click **OK** to save the new module. Press **Cancel** to return to the previous window.

Modifying Objects

You can modify objects at any time. You may choose to modify an object to incorporate a name change or any other change to the object. This section describes:

- Modifying Elements
- Modifying Structs
- Modifying Parameters

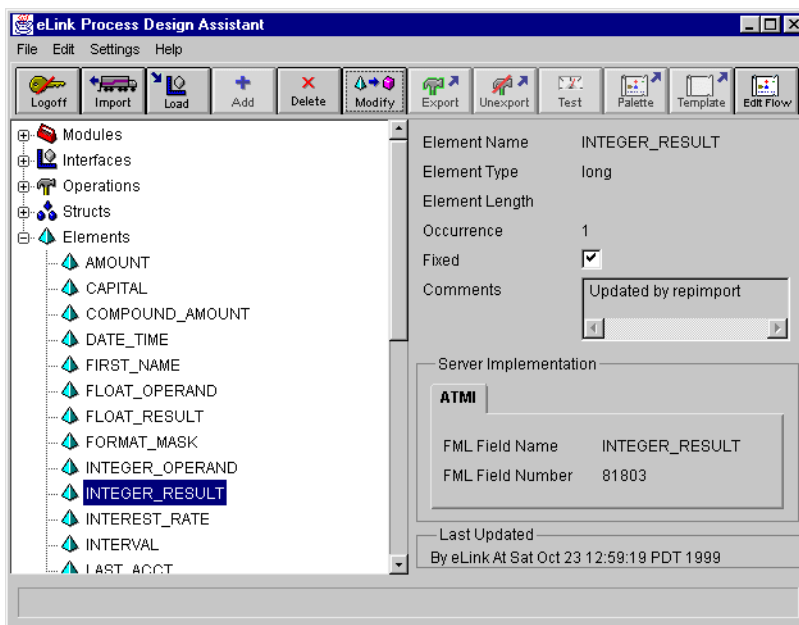
- Modifying Operations
- Modifying Interfaces
- Modifying Modules

Modifying Elements

You can modify an element at any time, including when you are creating new modules or after testing an operation. The Modify Element window in Figure 3-17 is used to modify elements. An element may be altered to incorporate occurrences, disable or enable the fixed-field length, add comments, and change server implementation information. See Adding Elements for a description of the element window.

Note: The element length can be altered only if the element type is string or carry.

Figure 3-16 Element Selection Window



Modifying Elements Instructions

To modify an element:

1. Select the element to modify.
2. Choose **Edit** —> **Modify** from the menu bar

Or

Click on the Modify toolbar button.

3. The Modify Element window shown in Figure 3-17 displays.

Figure 3-17 Modify Element Window

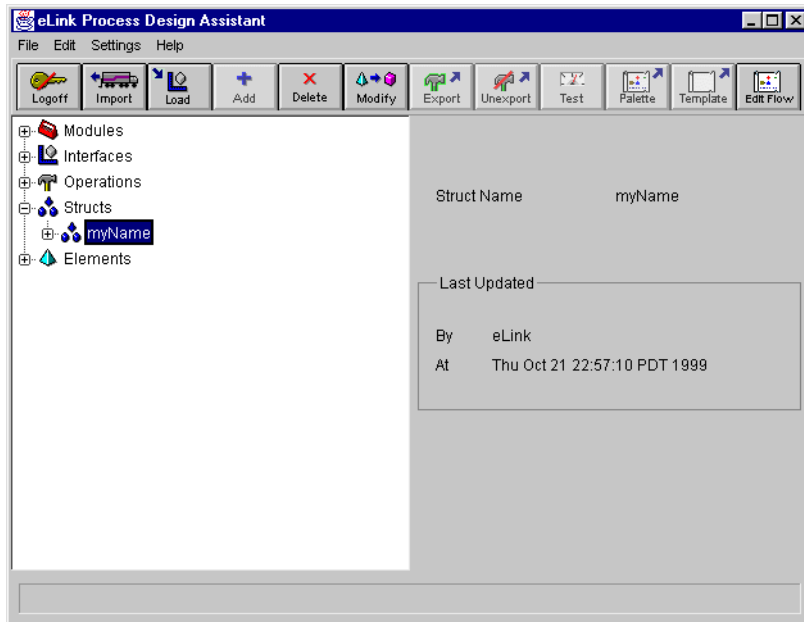
The screenshot shows a 'Modify Element' dialog box. The 'Element Name' field contains 'INTEGER_RESULT'. The 'Element Type' dropdown is set to 'long' with a 'Select' button. The 'Element Length' field is empty. The 'Occurrence' field contains '2'. The 'Fixed' checkbox is checked. The 'Comments' text area contains 'Updated by repimport'. Below this is a 'Server Implementation' section with a tab labeled 'ATMI'. Inside this section, the 'FML Field Name' dropdown is set to 'INTEGER_RESULT' with a 'Select' button, and the 'FML Field Number' field contains '81803'. At the bottom of the dialog, there is a 'Last Updated' section showing 'By eLink At Sat Oct 23 12:45:20 PDT 1999' and three buttons: 'OK', 'Cancel', and 'Help'.

4. Make the changes to the element.
5. Click **OK** to activate the changes and close the window.
(Or click **Cancel** to close the window without saving your changes.)

Modifying Structs

You can modify a struct from the Struct main window shown in Figure 3-18. To modify a struct, the current elements display in the left window and the available elements display in the right window as shown in Figure 3-19. The arrow buttons in the middle can be used to move elements in or out of the struct. See Adding Structs for a description of the struct window.

Figure 3-18 struct Selection Window



Modifying Structs Instructions

To modify a struct:

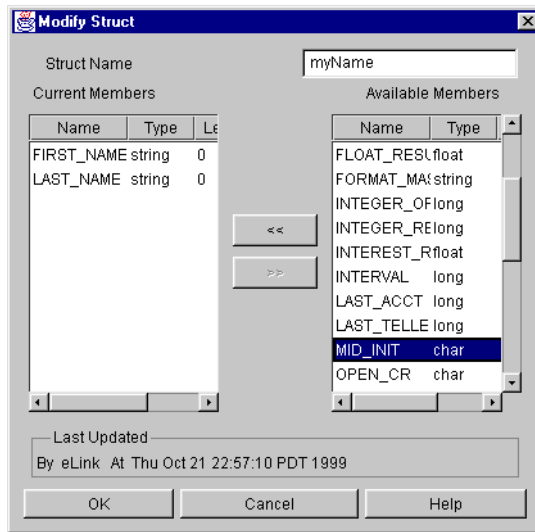
1. Select the struct to modify.
2. Choose **Edit** —> **Modify** from the menu bar

Or

Click on the Modify toolbar button.

3. The Modify Struct window shown in Figure 3-19 displays.

Figure 3-19 Modify Struct Window



4. Type the new struct name or move the members.
5. Click **OK** to activate the changes and close the window.
(Or click **Cancel** to close the window without saving your changes.)

Modifying Parameters

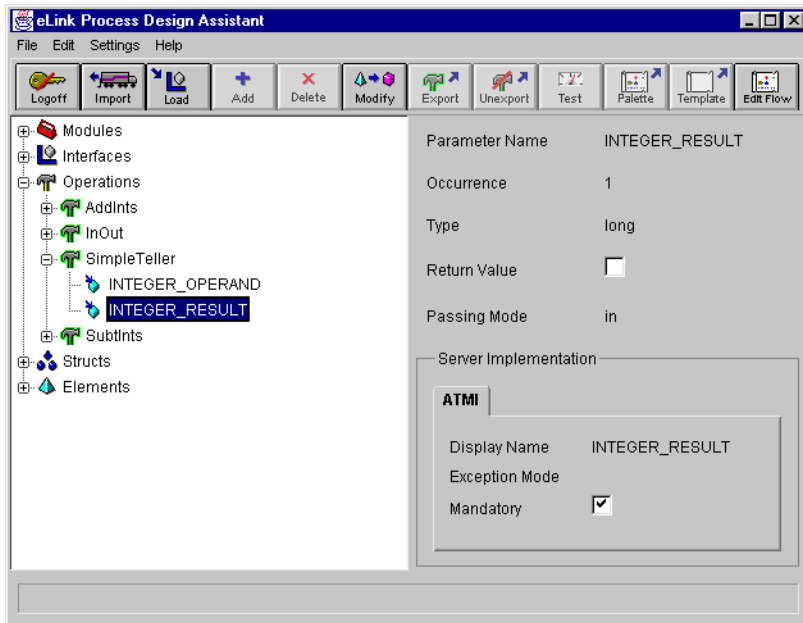
You can modify a parameter from the Business Interface Window (by selecting it in the tree and clicking **Modify** as shown in Figure 3-20).

Note: You can also modify a parameter as a part of adding or modifying operations. When you select a “current parameter” in the Add Operation or Modify Operation window and click **Edit**, you get the Edit Parameter window, which functions in the exact same way as the Modify Parameter window. (See Adding Operations or Modifying Parameters.)

For complete instructions see [Modifying Parameter Instructions](#). (The Modify Parameter window is shown in [Figure 3-21](#).)

A parameter may be modified to enable or disable the return value, or change the passing mode by selecting IN, IN/OUT, OUT, or NO ACCESS. In addition, the ATMI server information can be modified to reflect a display name change, a change in the exception mode return status, and enable or disable the mandatory status.

Figure 3-20 Parameter Selection Window



Modifying Parameter Instructions

To modify a parameter from the Business Interface Window:

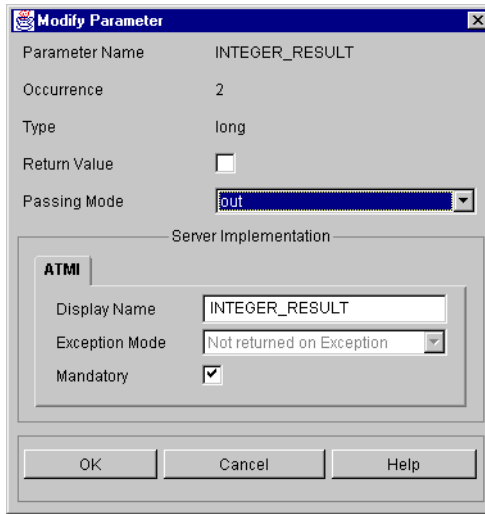
1. Select the parameter to modify.
2. Choose **Edit** → **Modify** from the menu bar

Or

Click on the Modify toolbar button.

This brings up the Modify Parameter window shown in Figure 3-21.

Figure 3-21 Modify Parameter Window



3. Fill in the fields as described in the following table.

Table 3-8 Modifying Parameters

Field	Description
Parameter Name	Indicates the same name of the element name used to create the parameter. To modify the parameter name, modify the element name first.
Occurrence	Indicates the number of occurrences of this element. If 0, the element can have an unlimited number of occurrences. The only way to modify the parameter occurrence is to modify the element occurrence. The occurrence number determines the number of data entry fields in the Test Operations dialog box. This is the only effect of this attribute.
Type	Type of element; either a primary element type FML (char, string, short, long, float, double, carray) or VIEW (int, short, long, char, float double, string carray) or a struct type. To modify the parameter type, modify the element type first.
Return Value	Select the checkbox if the element is used as a return value. Only one parameter from the Current Parameters list of an operation can have a return value.

Table 3-8 Modifying Parameters

Field	Description
Passing Mode	<p>Indicate the direction of movement for the data:</p> <ul style="list-style-type: none"> in out in/out noaccess <p>You can have only one parameter from the Current Parameters list for an operation that is listed as the return value. The passing mode of this parameter must be “out.”</p>
Display Name	Type a 32 character parameter name. This name must be unique within one operation.
Exception Mode	For out or in/out parameters, this mode indicates how the parameter is returned on exceptions. Enabled only for out and in/out passing modes when return exception is enabled for the operation.
Mandatory	Indicates if the parameter is required or optional. An occurrence of “1” is mandatory (if you selected the checkbox). An occurrence of “0” is not mandatory (if you did not select the checkbox).

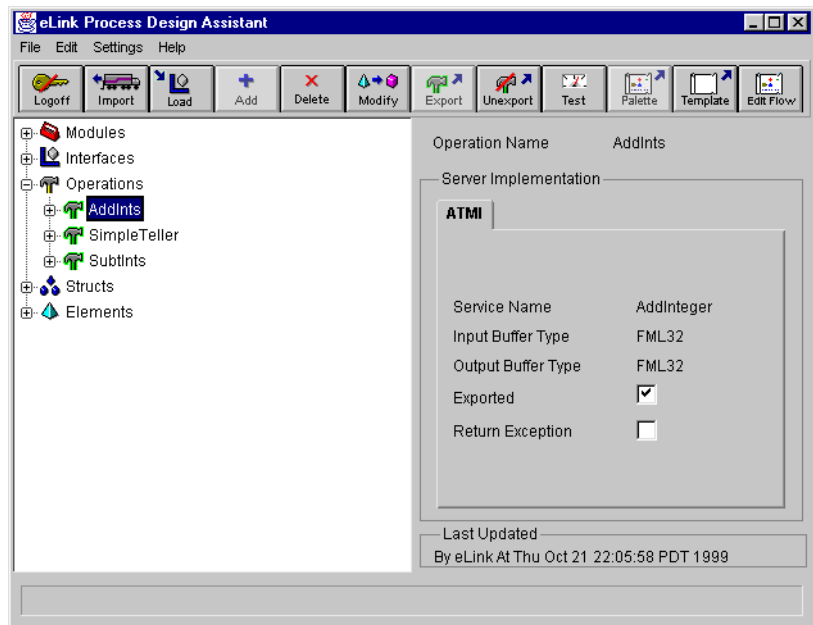
4. Make changes to the **Return Value**, **Passing Mode**, or **Display Name**.
 5. Click **OK** to activate the changes and close the window. When the changes have been made to the module, the update status changes to reflect the most recent modification.
- (Or click **Cancel** to close the window without saving your changes.)

Modifying Operations

You can modify an operation by selecting an operation from the repository tree as shown in Figure 3-22. See Adding Operations for a description of the operation window.

Note: You can modify the components of an exported operation. You do not have to unexport the operation in order to modify its components. If you modify an exported operation, it does not automatically change to *unexported* status. A modified exported operation remains exported until you unexport it with the unexport toolbar button or menu option.

Figure 3-22 Operation Selection Window



Modifying Operations Instructions

To modify an operation:

1. Select the operation to modify.
2. Choose **Edit** —> **Modify** from the menu bar

Or

Click on the Modify toolbar button.

3. The Modify Operation window in Figure 3-23 displays.

Figure 3-23 Modify Operation Windows

Modify Operation

Operation Name: AddInts

Current Parameters			Available Parameters		
Name	PassingMode	Length	Name	Type	Length
INTEGEFin		0	AMOUNT	float	0
INTEGEFout		0	CAPITAL	float	0
			COMPOUND_float	float	0
			DATE_TIME	string	0
			FIRST_NAME	string	0
			FLOAT_OPEF	float	0

Server Implementation: ATMI

Service Name: AddInteger

Input Buffer Type: FML32

Output Buffer Type: FML32

Exported Return Exception

Last Updated: By eLink At Thu Oct 21 22:05:58 PDT 1999

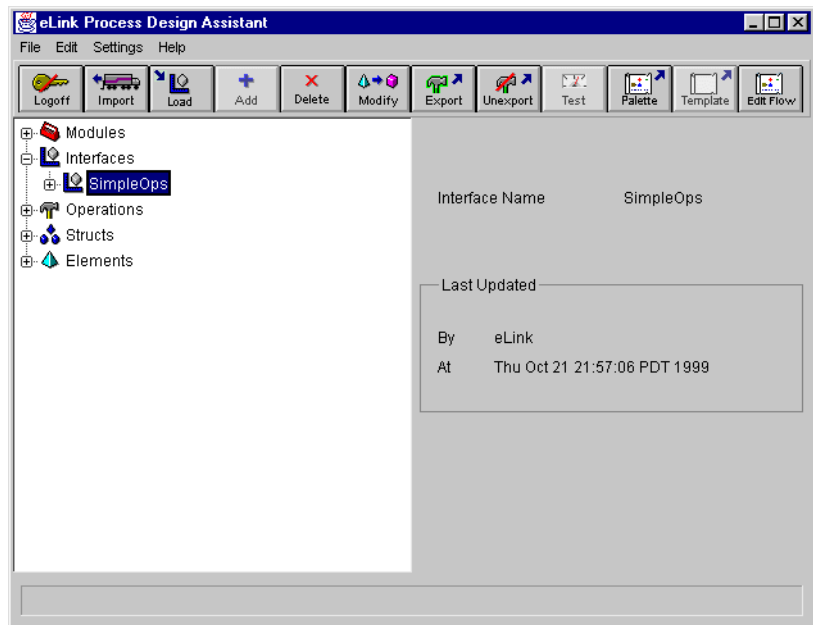
OK Cancel Help

4. Type the new operation name, move the parameters, edit parameters, or modify the **Server Implementation** details.
5. Click **OK** to activate the changes and close the window.
(Or click **Cancel** to close the window without saving your changes.)

Modifying Interfaces

You can make changes to an interface by selecting the interface from the directory tree shown in Figure 3-24. See Adding Operations for a description of the interface window.

Figure 3-24 Interface Selection Window



Modifying Interfaces Instructions

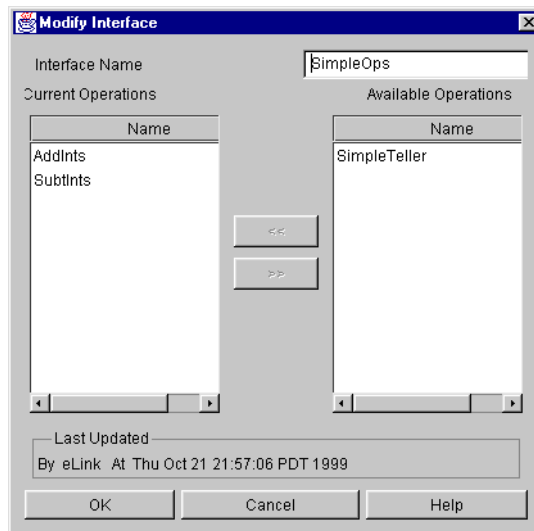
To modify an interface:

1. Select the interface to modify.
2. Choose **Edit** —> **Modify** from the menu bar

Or

Click on the Modify toolbar button.

3. The Modify Interface window in Figure 3-25 displays.

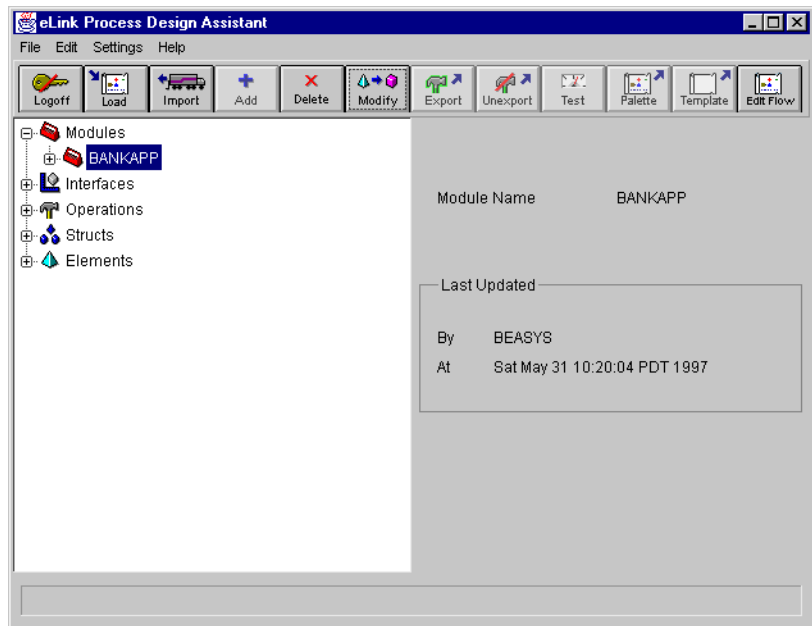
Figure 3-25 Modify Interface Window

4. Use the << or >> keys to move the operations.
5. Click **OK** to make the changes. When the changes have been made to the module, the update status changes to reflect the most recent modification.
(Or click **Cancel** to close the window without saving your changes.)

Modifying Modules

Modify a module by selecting the object from the directory tree shown in Figure 3-26. See Adding Operations for a description of the module window.

Figure 3-26 Modules Selection Window



Modifying Modules Instructions

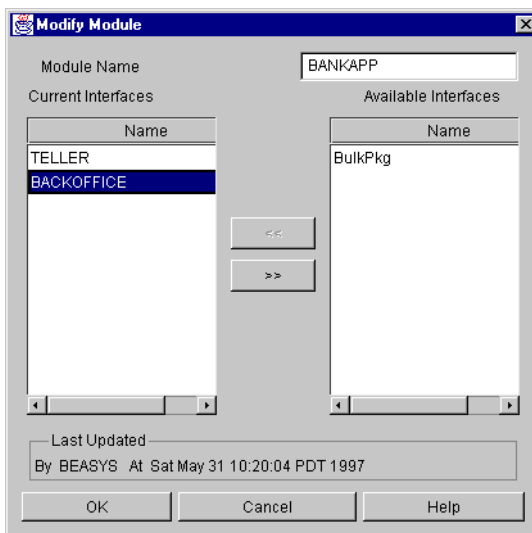
To modify a module:

1. Select the module to modify.
2. Choose **Edit** —> **Modify** from the menu bar

Or

Click on the Modify toolbar button.

3. The Modify Module window in Figure 3-27 displays.

Figure 3-27 Modify Module Window

4. Use << or >> keys to move the interfaces between the **Current Interfaces** and **Available Interfaces** display lists.
5. Click **OK** to make the changes. When the changes have been made to the module, the update status changes to reflect the most recent modification.
(Or click **Cancel** to close the window without saving your changes.)

Deleting Objects From the Repository

This section details the necessary steps to delete an object. You can delete a module, interface, operation, parameter, struct, or element. After deletion, the object is gone. There is no recycle bin in the Contract Repository, so if you inadvertently delete an object you must create a new one to replace it.

You cannot automatically delete an object that is dependent on (or used by) another object. If you attempt to delete an object that is in use in the Contract Repository, you receive an Alert message. The Alert message shows where the object is being used. When you click **OK**, the Alert window closes. So, before you close the Alert window, you may want to note where the object is being used. You cannot delete the object until you remove the dependency.

A module is the only object that does not have any dependencies.

Deleting Instructions

To delete an object:

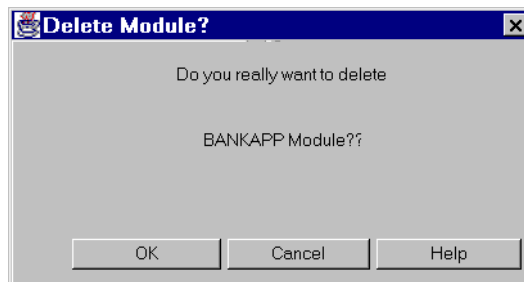
1. Select the object to be deleted from the Business Interface Window.
2. Choose **Edit** —> **Delete** from the menu bar

Or

Click on the Delete toolbar button.

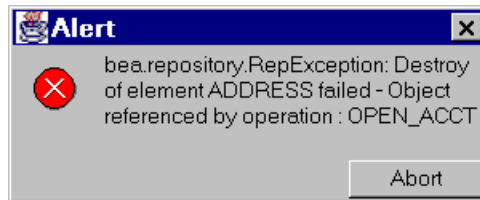
3. To confirm that you want to delete the object, the dialog box, “Do you really want to delete <name of the object>?” displays.

Figure 3-28 Delete Module Window



4. Click **OK** to delete. Click **Cancel** to return to the previous window.
5. If there is a dependency, the Alert window displays. If you still want to delete the object, refer to the Alert window shown in Figure 3-29 for the dependencies.

Figure 3-29 Alert Window with Dependency Information



6. In this example, the object, ADDRESS element, cannot be deleted because it is referenced by (or is depended upon by) the OPEN_ACCT operation. If you wish to delete the object, you must remove the dependency by modifying the OPEN_ACCT operation. In this case, you would need to remove the ADDRESS element from OPEN_ACCT before proceeding with the delete. If you have to remove a parameter, modify the operation and delete it using the Modify Operation window.

For additional information, refer to the Modifying Objects section in this chapter.

Exporting or Unexporting an Operation

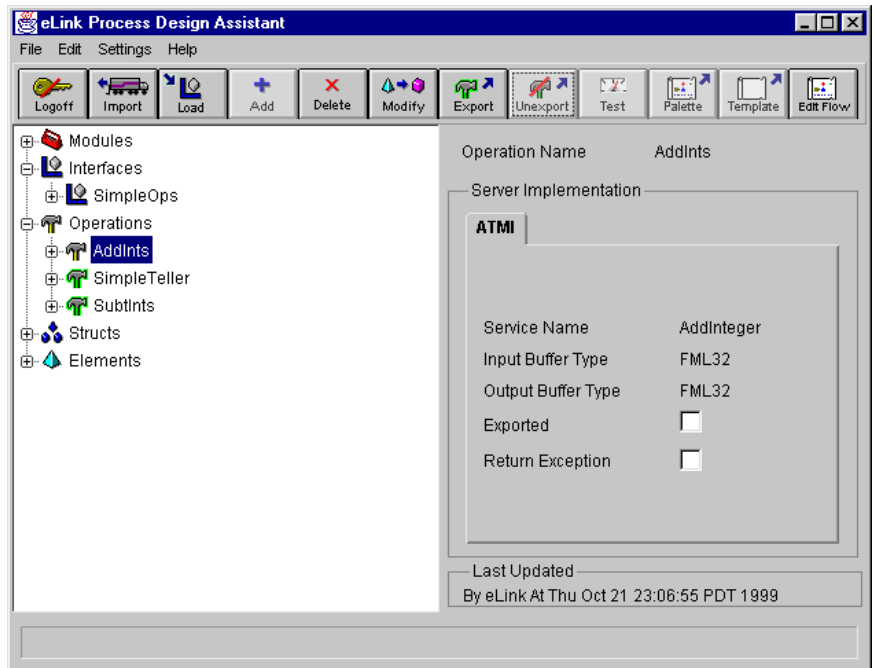
Once you have completed building an operation, you are ready to test it. Before you can test an operation, you must export it. “Export” serves as a flag that makes it available for testing, generating palettes and templates, and web or Windows applications for use in run time.

By default, the operation is always unexported as shown in Figure 3-30.

You have the option to export or unexport all operations within the interface at one time.

You do not have to make all operations available to the client at the same time, even if your interface contains several operations.

Figure 3-30 Window with Unexported Operation



Export Instructions

To export:

1. Select an interface or operation from the Business Interface Window. If you select an interface, all of the operations in the interface will be exported.

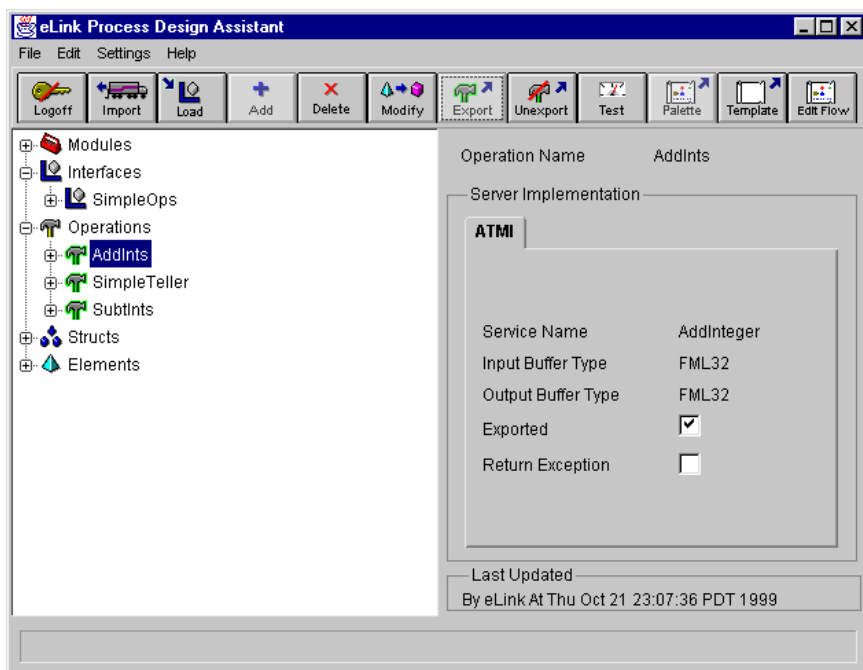
2. Choose **Edit** → **Export** from the menu bar

Or

Click on the Export toolbar button.

3. The **Exported** checkbox is checked to indicate that the operation has been exported as shown in Figure 3-31.

Figure 3-31 Window with Exported Operation



Unexport Instructions

To unexport an operation:

1. Select an interface or operation from the Business Interface Window. If you select an interface, all operations in the interface will be unexported.

2. Choose **Edit** —> **Unexport** from the menu bar

Or

Click on the Unexport toolbar button.

3. The **Exported** checkbox is unchecked to indicate that the operation has been unexported as shown in Figure 3-30.

Testing an Operation

You can test an operation and its parameters to ensure that all components are functioning properly before you can use the operation further.

Once an operation is exported you can test it. If the test fails and editing is required to fix the operation, you do not have to unexport the operation prior to editing.

The operation test window allows you to test previously defined operations to verify their function against the service implementation. The window contents are based on the number and type of parameters the operation expects. If an operation has multiple parameters and cannot be viewed on one screen, a scrollbar displays to navigate through the parameters.

Figure 3-32 Test Operation Window

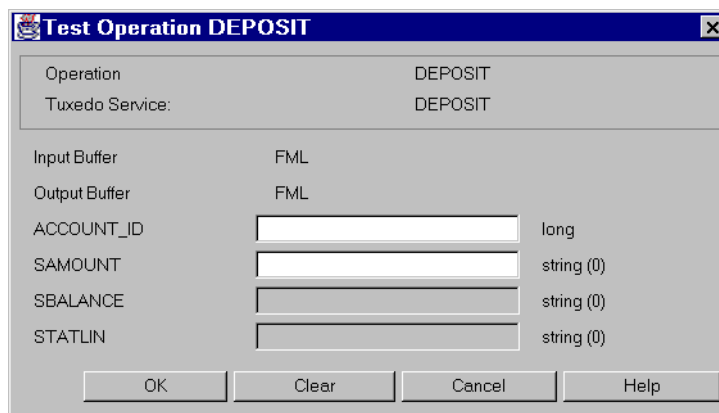


Table 3-9 Test Operation

Field	Description
Operation	The name of operation that you selected to be tested.
ATMI Service	The name of the associated ATMI service.
Input Buffer and Output Buffer	Only FML32 buffer types are supported. FML32 — a type in which each field carries its own definition. NONE — placeholder indicating that no input or output is required.

Table 3-9 Test Operation

Field	Description
Parameter text fields	The parameter information text entry field. The fields that display in the test window are dependent on the operation you are testing. Based on the passing mode, these fields are writable or read-only. Fields are disabled if read-only.
OK	Runs the operation.
Clear	Clears all fields and resets the scrollbar to the top (if a scrollbar is present).
Cancel	Exits the window.

Guidelines for Inputting Data

- Do not leave mandatory input parameters blank.
- Do not leave mandatory in/out parameters blank.
- If the parameter fields are fixed, all occurrence fields must be filled in.
- You can leave all or some of the occurrence fields blank if the parameter field is not fixed.

Test Operation Instructions

To test an operation:

1. Select the operation to test.
2. Choose **Edit** —> **Test** from the menu bar
Or
Click on the Test toolbar button.
3. Input data in the Operation test window parameter text field.
4. Click **OK**. The message, “Test Complete,” displays if the test passes. If the test fails, a message displays in the Alert window.

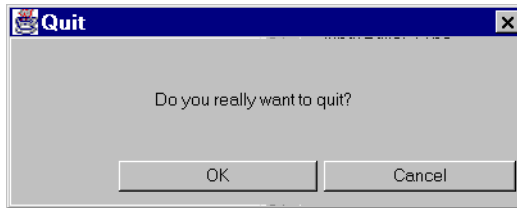
Follow the instructions below if editing is required to pass the test.

- a. Return to the Business Interface Window and select the operation, parameter, struct or element that requires editing.
- b. Choose **Edit**—>**Modify** from the menu bar.

Logging Off and Exiting

Log off the Process Design Assistant before exiting it. The Logoff option is only enabled after logon.

Figure 3-33 Exit Prompt



Logoff/Exit Instructions

To log off and exit:

1. To log off the Process Design Assistant, choose **File** —> **Logoff** from the menu bar
Or
Click on the Logoff toolbar button.
2. The opening Process Design Assistant window displays.
3. To exit the Process Design Assistant, choose **File**—> **Exit** from the menu bar.
4. The prompt, “Do you really want to quit?” displays in the dialog box shown in Figure 3-33. Click **OK** to exit or **Cancel** to return to the Process Design Assistant opening window.

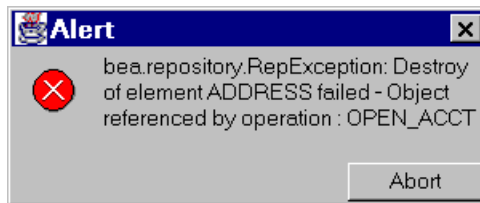
Troubleshooting

You are “alerted” through an alert window of any error that occurs while running the application. You will receive an alert message if the system is unable to:

- Connect to the server
- Read the Contract Repository
- Understand the data received from the Contract Repository
- Write to disk
- Validate information that you entered

The message in Figure 3-34 is an alert about an object dependency.

Figure 3-34 Alert Window



Alert Window Instructions

When the Alert window displays:

1. Read the message and follow the suggestions for a possible solution.
2. Click **Abort** upon completion of reading the message to remove the window.

Process Design Assistant and ATMI Mappings

Table 3-10 Mappings Between Contract Repository and ATMI

Contract Repository	ATMI
module	—
interface	—
operation	service
structure	Group of FML fields
exception	TPFAIL with return buffer
parameter	FIELD in buffer
parameter (return value)	FIELD in buffer
mandatory occurrence	FIELDS in buffer
maximum occurrence	FIELDS in buffer
element	FIELD definition in FML field table

4 Designing Business Processes

Overview

This chapter discusses the designing of process flows using the Process Design Assistant. The following topics are covered in this chapter:

- Overview
- How It Works
- Usage
- Using the Business Process Window
- Working with Process Definitions
- Working with Tasks
- Specifying attributes

How It Works

The Process Design Assistant Business Interface Window is an eLink client/server application. Contracts are specified in the Business Interface Window (client) and stored in the Contract Repository database (server). For more information on

specifying contracts, see Chapter 3, “Specifying Business Service Contracts.” Palettes and templates are then generated from these contracts and appear in the Business Process Window. The Business Process Window is the subject of this chapter.

Operations are converted into tasks during these palette and template generation processes. Operations are only definitions of inputs and outputs. Tasks are more than a definition of inputs and outputs. They have other properties and attributes that allow them to be executed by the Business Process Engine at runtime. (These properties and attributes will be discussed in Chapter 5, “Making eLink Processes Work.”) These generated tasks are sequenced with dependencies using the tools within the Business Process Window. The sequenced tasks make up a process flow.

Let us introduce the concept of palettes and templates with an analogy. A palette has a similar function to a painter’s palette. Just as a painter fills his palette with the appropriate colors for his painting, we fill our palette with the appropriate tasks for our process flow. A template is like a painter’s canvas. Just as he will size and shape the canvas to fit the painting, we must specify certain characteristics of the template we will use for a particular process flow. Once we have done this, we are ready to “paint” our process flow by copying tasks from our palette to our template. Finally, as a painter might choose to finish his painting by creating just the right frame for it, we too have finishing steps for our process flow. These are discussed in Chapter 5, “Making eLink Processes Work.”

Usage

This section provides instructions for generating palettes and templates from contracts and forming the tasks created from those contracts into process flows.

Generating a Palette

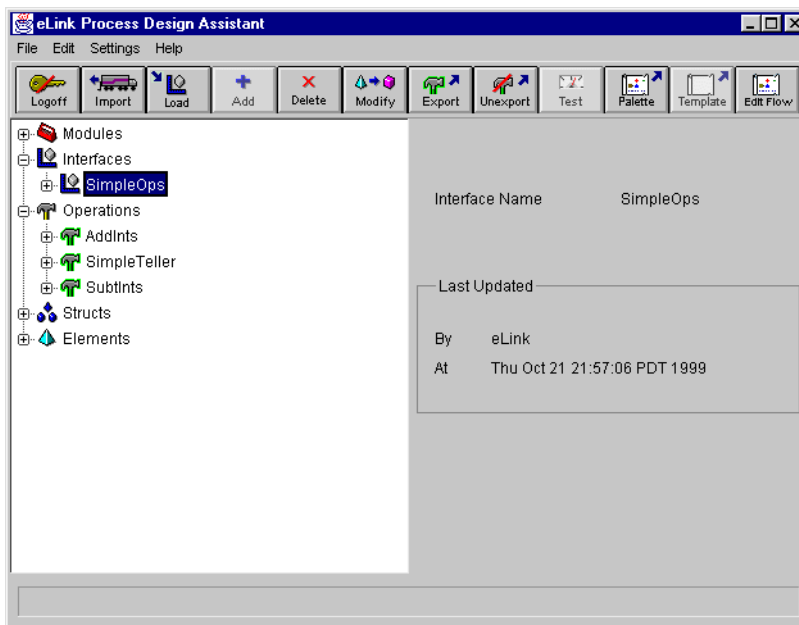
Once you have completed building an interface in the Process Design Assistant Business Interface Window, have exported and then tested all of the operations it contains, you are ready to generate a palette from it. Generated palettes appear in design pads where the tasks they contain can be used to form process flows.

Instructions for Generating a Palette

To generate a palette from an interface:

1. Select an interface from the Business Interface Window.

Figure 4-1 Business Interface Window with Interface Selected



All of the operations contained in the selected interface will be available (and appear on the palette) for use in forming process flows. For example, if you generate a palette from the SimpleOps interface (selected in Figure 4-1), the operations it contains will appear as tasks on the palette.

2. Choose **Edit** —> **Generate Palette** from the menu bar

Or

Click on the Palette button.

The Palette Generation Results dialog appears, and a log of the palette generation actions displays in the dialog box.

Generating a Template

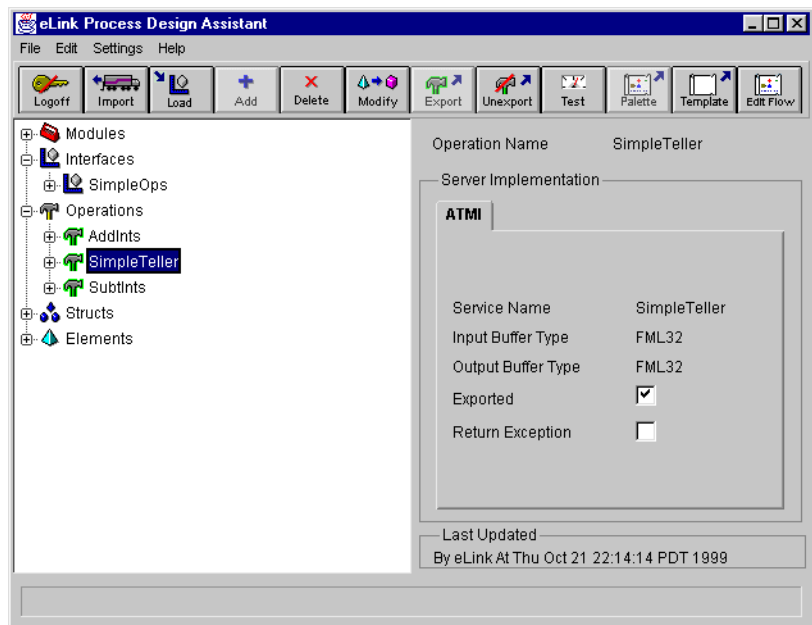
Once you have generated a palette containing the tasks that you want to use in your process flow, you must generate a template from an operation in the Business Interface Window. This template will be the container for your process flow.

Instructions for Generating a Template

To generate a template:

1. Select an exported and tested operation from the Business Interface Window.

Figure 4-2 Business Interface Window with Operation Selected



2. Choose **Edit** —> **Generate Template** from the menu bar

Or

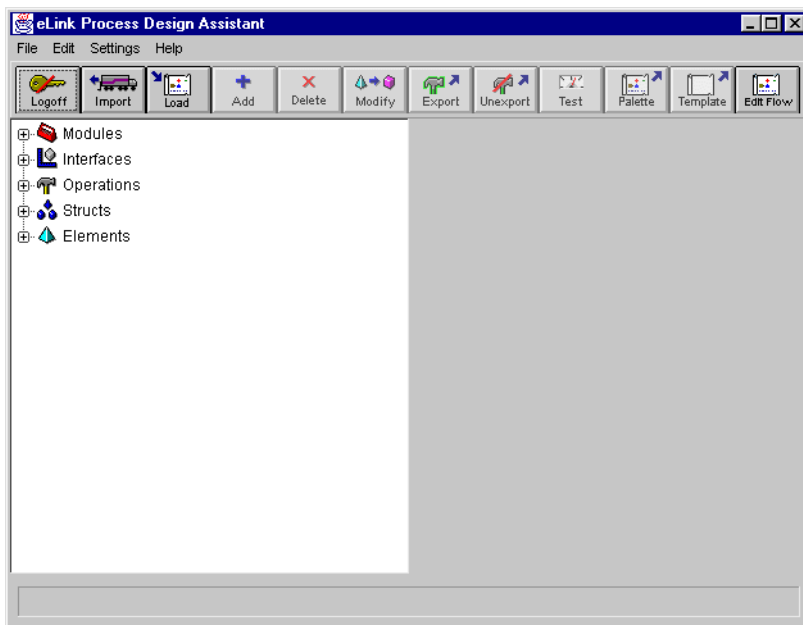
Click on the Template button.

The Template Generation Results dialog appears, and a log of the template generation actions displays in the dialog box.

Opening the Business Process Window

Once you have generated a palette and a template, you must open the Business Process Window so you can form your process flow.

Figure 4-3 Main Window with Edit Flow button



Instructions for Opening the Business Process Window

To open the Business Process window:

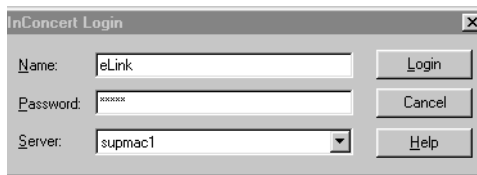
1. Choose **Edit** —> **Edit Flow** from the Business Interface Window menu bar

Or

Click on the Edit Flow button.

The Login Confirmation Dialog Box appears, shown in Figure 4-4.

Figure 4-4 Login Confirmation dialog box



Confirming Login to the Business Process Database

When the Business Process Window opens, you must confirm your login to the Business Process Database. The Login Confirmation Dialog Box will be filled in based on the Business Interface Window login. Click **Login** to confirm.

Using the Business Process Window








After the login is confirmed, the login dialog box closes and the Business Process Window appears.

The toolbar provides menu shortcuts. Every button on the toolbar has a corresponding menu option. The buttons on the left side of the tool bar provide access to Business Process Window applications.

The *status bar* at the bottom of the Business Process Window displays system messages and abbreviated help text.

Table 4-1 shows the buttons that are on the toolbar as you work in the Business Process Window.

Table 4-1 Design Pad toolbar buttons

Button	Action
	Create a new process definition.*
	Open an existing process definition.
	Change displayed attributes for a task.
	Select or move a task or dependency.
	Create a new task.*
	Create a new dependency between tasks.
	Open the Task User Interface Designer.**

Note: *Tasks and processes must be created as part of the palette and template generation processes to be functional in eLink. The Business Interface Window provides the means for generating these palettes and templates. You can create new tasks and processes with these buttons, but they cannot be used in eLink.

**The Task User Interface Designer is not used in eLink.

Using the Business Process Window

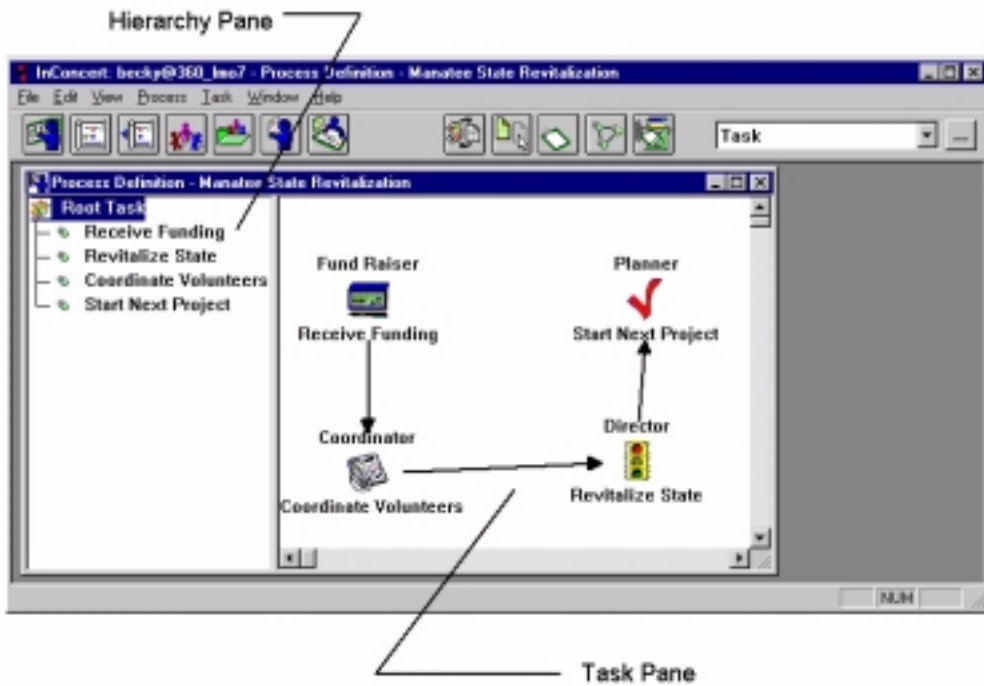
This section explains how to use and navigate the Business Process Window. The Business Process Window is used for manipulating processes and the tasks that comprise them. You can modify an existing process definition or modify an active process using the Business Process Window.

Getting started with the Business Process Window

To use the Business Process Window, you must open an existing process definition. The process definition opens on a Design Pad.

The Design Pad window consists of the *hierarchy pane* on the left, and the *task pane* on the right, as shown in Figure 4-5.

Figure 4-5 Design Pad window



The hierarchy pane displays the structure of the process as a whole. When you select a task in the hierarchy pane, its Child tasks (if any) appear in the task pane.

If a task has a subtask, a bracketing icon appears in the hierarchy pane, shown in Figure 4-6.

Figure 4-6 Icon in hierarchy pane indicating subtasks



If a task has a subtask, a blue triangle icon appears in the task pane next to the task name, shown in Figure 4-7.

Figure 4-7 Icon in task pane indicating subtasks

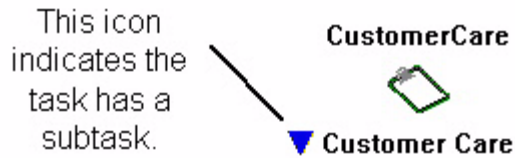
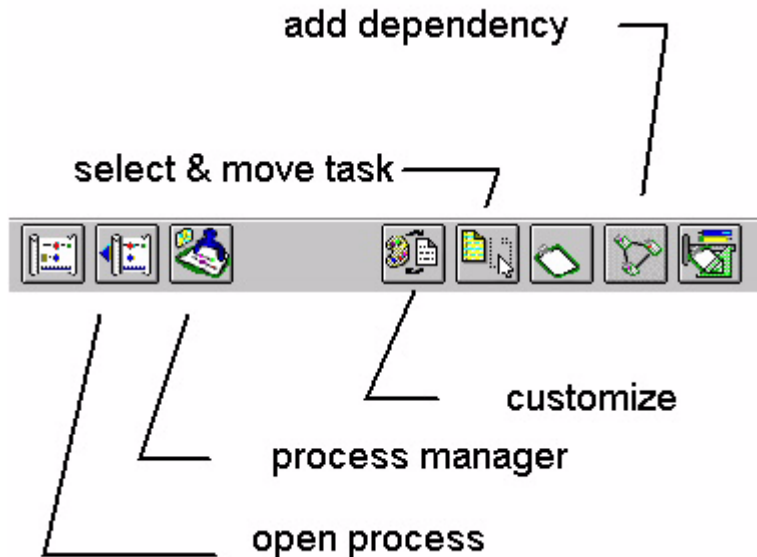


Figure 4-8 shows the buttons that appear on the Business Process window toolbar and what those that are used by eLink do.

Figure 4-8 Buttons on the Process Designer toolbar



The remainder of this chapter explains how to work with the Business Process Window and its sub-windows once the Business Process Window is open.

Refreshing the Design Pad display

Occasionally, the Design Pad does not show your most recent change clearly. If this occurs, you can refresh the display.

Instructions for Refreshing the Design Pad

To refresh the Design Pad:

Press the **F5** key, or choose **Window > Refresh**.

Printing from the Design Pad

You can print the contents of the hierarchy and task panes.

The first page of the printout shows the contents of the hierarchy pane. The printout includes only visible (expanded) tasks; for example, the root task and its immediate children.

The second page shows the graphical representation of the tasks in the task pane. The print out includes all tasks in the task pane, even if some of the tasks are not currently visible (that is, they are scrolled off screen).

In the printout, task icons are represented by boxes. Task names and attributes are displayed above and below each icon box, just like in the task pane. Dependency arrows are also shown.

The task layout is printed on multiple pages if it does not fit on a single page. The pages are organized from left to right and top to bottom.

Instructions for Printing the Design Pad Display

To print the Design Pad display:

Choose **File > Print**.


Processes are automatically saved

You can now proceed with other activities immediately because changes in the Business Process Window are saved as you make them. There is no need to explicitly save your work. This ensures that the structural elements of the process are always up-to-date in the database, and that attribute settings are correct.

If you have more design work to do, keep the Business Process Window open. Otherwise, you can close the Business Process Window.

Instructions for Closing the Business Process Window

To close the Business Process Window:

Click the  in the top left corner of the process map window.

Defining and changing process structure

Most design activity involving the process structure takes place within the task pane of the Design Pad. Use your mouse to perform the following design and layout operations:

- Add dependencies
- Select and move tasks

You can also copy tasks, and delete tasks and dependencies.



You can switch between layout operations, and “jump” temporarily from one operation to another.

Table 4-2 shows a summary of the design and layout operations.

Table 4-2 Summary of Design Pad operations

To...	First...	Then...
Select a task	Choose Edit > Select and Move	Click left on the task.
Move a task	Choose Edit > Select and Move	Press and hold the left mouse button and drag task to new location. To move a task while you are in another layout mode, press and hold the Ctrl key and click right.
Copy a task	Choose Edit > Select and Move	Select task, then choose Edit > Copy . Select new parent task and choose Edit > Paste .

Table 4-2 Summary of Design Pad operations (Continued)

To...	First...	Then...
Add a dependency	Choose Edit > Create Dependencies	Click left on precedent task, drag to dependent task, and release. To move a task while you are in another layout mode, press and hold the Ctrl and Shift keys and click right.
Delete a task or dependency	Choose Edit > Select and Move	Select task or dependency and press the Delete key.
Add a PerformCondition	Choose Edit > Select and Move	Select task, then choose View > Properties . Select the Perform Condition attribute's value and click the  button.
Add an IterateCondition	Choose Edit > Select and Move	Select task, then choose View > Properties . Select the Iterate Condition attribute's value and click the  button.
Delete a PerformCondition	Choose Edit > Select and Move	Select the task, then choose View > Properties . Delete contents of PerformCondition attribute.
Delete an IterateCondition	Choose Edit > Select and Move	Select the task, then choose View > Properties . Delete contents of IterateCondition attribute.

The section Working with Tasks describes in detail each task operation you can perform in the Design Pad.

Navigating the process structure

To keep the display manageable, the Design Pad shows only selected parts of the process structure. When you select a task in the hierarchy pane, only its immediate subtasks display in the task pane.

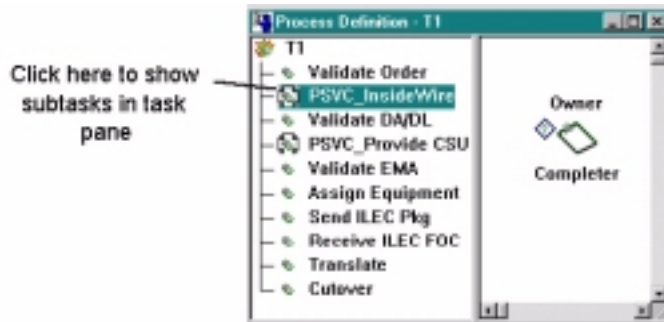
You can navigate the process structure from the hierarchy pane, the task pane, or a combination of both. From the hierarchy pane, you can navigate the process structure and show and hide different parts of the process. From the task pane, you can go only to the next lower level of the process.

Instructions for Navigating from the Hierarchy Pane

To navigate from the hierarchy pane:

1. In the hierarchy pane, select the task whose subtasks you want to see. The subtasks appear in the task pane, shown in Figure 4-9. If the task has no subtasks, nothing is displayed in the task pane.

Figure 4-9 A task with a subtask in the task pane




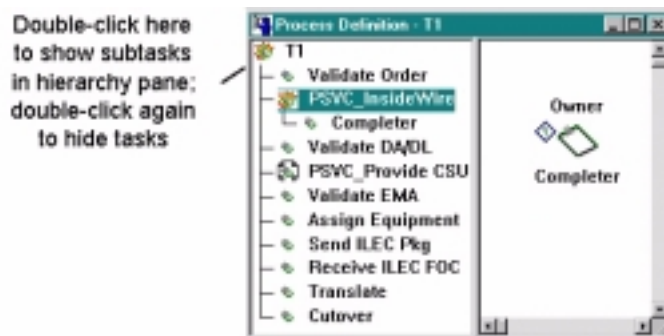
2. To view the task's subtasks in the hierarchy pane as well as the task pane, double-click the task name or select the task and choose **View > Expand**. The icon next to the task name changes to  indicating that the task is expanded. Figure 4-10 shows an example of an expanded hierarchy pane. When you expand any task (including the Root Task) in the hierarchy pane, its subtasks are displayed in the task pane.

Figure 4-10 A task with a subtask in the hierarchy pane



Instructions for Hiding a Task's Children in the Hierarchy Pane

To hide a task's children in the hierarchy pane:

Double-click the task name, or select the task and choose **View > Collapse**. The icon next to the task name changes back to the standard icon.

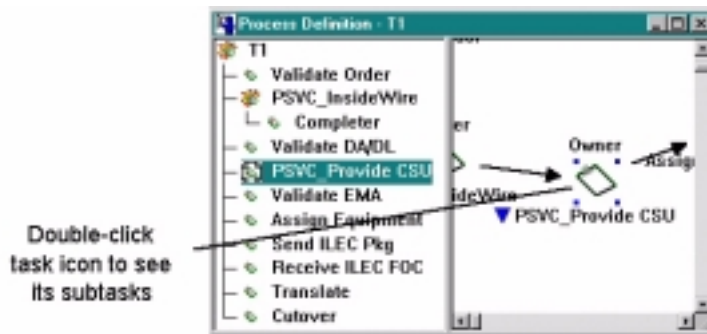
Note: Collapsing a task in the hierarchy pane often has no effect on the task pane. For example, if you collapse the PSVC_InsideWire task in Figure 4-10, the task pane is unchanged. However, if you collapse a task several levels above that task (for example, the T1 task), the task pane displays the subtasks of that newly selected task.

Instructions for Navigating from the Task Pane

To navigate from the task pane:

1. Choose **Edit > Select and Move**, or click the **Select and Move** button on the toolbar.
2. Double-click the task's icon to display its subtasks. The task you double-clicked is outlined in the hierarchy pane and its child tasks are displayed in the task pane. Figure 4-11 shows an example of navigating in the task pane.

Figure 4-11 Navigating in the task pane



Instructions for Navigating from the Property Sheet

To navigate from the Property Sheet:

Click the Task tab of the Property Sheet. Select a task in the **Task Name** listbox. The Design Pad display moves and centers on that task.

Working with multiple Design Pad windows

You can open multiple Design Pad windows and work on different parts of the same process or edit several different processes at once. The advantages of working with multiple Design Pad windows are that you can:

- Copy and paste tasks between each window
- Drag and drop tasks from one task pane to another
- View the process structure independently in each Design Pad window
- Compare attributes for different processes

You can open as many additional processes as you want. The Property Sheet tracks task selections only in the active Design Pad.

Instructions for Working on Different Parts of the Same Process



To work on different parts of the same process:

Choose **Window > New Window**. The process opens in another Design Pad and any changes you make in one Design Pad are immediately reflected in the other.

You can open the process in as many Design Pads as you like. The Property Sheet is shared among Design Pads, and tracks task selections only in the active Design Pad.

Instructions for Working on a Different Process

To work on a different process:

Choose **File > New > Process Definition** or **File > Open > Process Designer** (see for details), or click the  or  buttons on the toolbar.

Copying tasks between Design Pad windows

The procedure for copying tasks between Design Pad windows is similar to copying tasks in a single Design Pad.

Instructions for Copying Tasks Between Design Pad Windows

To copy tasks between Design Pad windows:

1. Select a task in the task pane of one Design Pad.
2. Choose **Edit > Copy**.
3. Go to the other Design Pad and select the new parent task in the hierarchy pane.
4. Choose **Edit > Paste**.
5. Adjust the task layout, as needed.

Using drag and drop between Design Pad windows

You can drag and drop tasks between Design Pad windows as an alternative to cut, copy, and paste. You can drag and drop tasks from one part of a process to another or between processes.

Note: Drag and drop within a process moves the tasks from one part of the process to another. Drag and drop between different processes copies the tasks from one process to another.

There are two ways to drag and drop tasks from one Design Pad to another. The method you choose depends on whether the task pane in the destination Design Pad contains the new parent task. The new parent task does not have to be *visible* in the task pane, however; when you drag and drop tasks in the Design Pad, the task pane scrolls automatically when you reach the window edge.

Instructions for Dragging and Dropping a Task When the Destination Task Pane Contains the New Parent Task

To drag and drop a task when the destination task pane contains the new parent task:

1. Select a task in the task pane on one Design Pad. Do not release the mouse button after selecting the task. The cursor changes to a drag icon.

2. Drag the task to the destination Design Pad and drop it on the icon of the new parent task. A popup menu is displayed from which you can choose the kind of operation you want to perform.
3. Choose one of these options from the pop-up menu:
 - Choose **Create Subtask** to insert the task and its descendents into the process structure.
 - Choose **Cancel** to cancel the operation.

Note: Under some conditions, the **Paste Interface** option may also appear. This option does not apply to eLink.

If the Design Pad contains the same process, the task and its descendents *move* to the new location. If the Design Pad contains a different process, the task and its descendents are *copied* into the process structure.

Instructions for Dragging and Dropping a Task When the Destination Task Pane Does Not Contain the New Parent Task

To drag and drop a task when the destination task pane does not contain the new parent task:

1. On the destination Design Pad, select the new parent task in the *hierarchy* pane.
2. Go to the other Design Pad and select a task in the task pane, but do not release the mouse button. The cursor changes to a drag icon.
3. Drag the task to the destination Design Pad and drop it on the task pane background (any open area in the task pane).
4. If the Design Pad contains the same process, the Business Process Engine *moves* the task and its descendents to the new location. If the Design Pad contains a different process, the Business Process Engine *copies* the task and its descendents into the process structure.
5. Adjust the task layout, as needed.

Working with Process Definitions

This chapter explains how to use process definitions in your process flow design.

To do this, you must open an existing process definition (template) and make modifications (add tasks from a palette, dependencies between them, parameter assignments, and PerformConditions) to create a process flow.

This section covers this method for forming a process flow. The section of this chapter, Working with Tasks, covers procedures for adding and working with tasks during process design.

Working with existing process definitions

You can modify an existing process definition and save it under its original name.

Note: A template generated within the Business Interface Window is the only type of process definition that can be formed into a working eLink process flow.

Instructions for Preparing to Form a Process Flow

To prepare for forming a process flow:


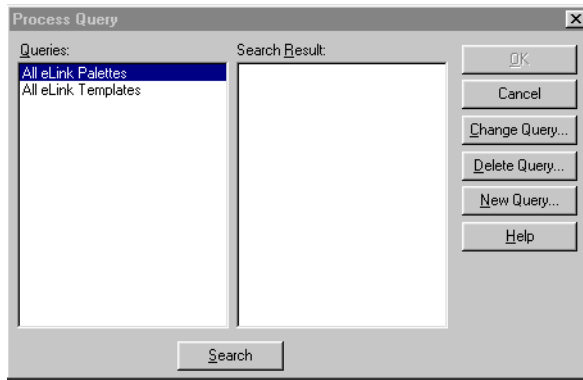
1. Select **File > Open > Process Designer**, or click the  button on the toolbar. The Process Query dialog box appears, shown in Figure 4-12. You must use a query to search for the process definition (template or palette).

Figure 4-12 The Process Query dialog box

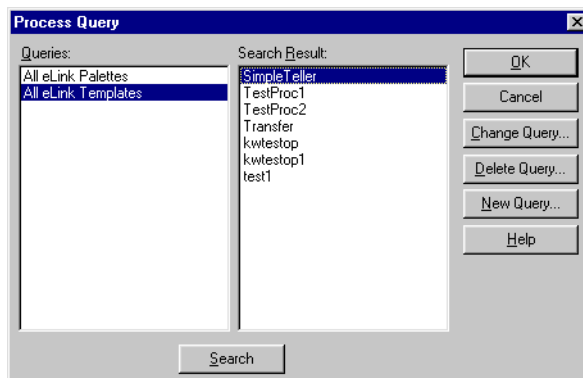


2. Select the All eLink Templates query from the **Queries** listbox.

The **Queries** listbox will contain all of the templates generated from the Process Design Assistant Business Interface Window. If no templates appear in the listbox, you must generate one by returning to the Process Design Assistant Business Interface Window and generate a template according to the instructions in the Generating a Template section of this chapter.

3. Click **Search**, or double-click the query name. The process definitions (templates) found by the query are displayed in the **Search Results** listbox, shown in Figure 4-13.

Figure 4-13 Search results in the Process Query dialog box



4. In the **Search Results** listbox, select the process you want to open for editing.

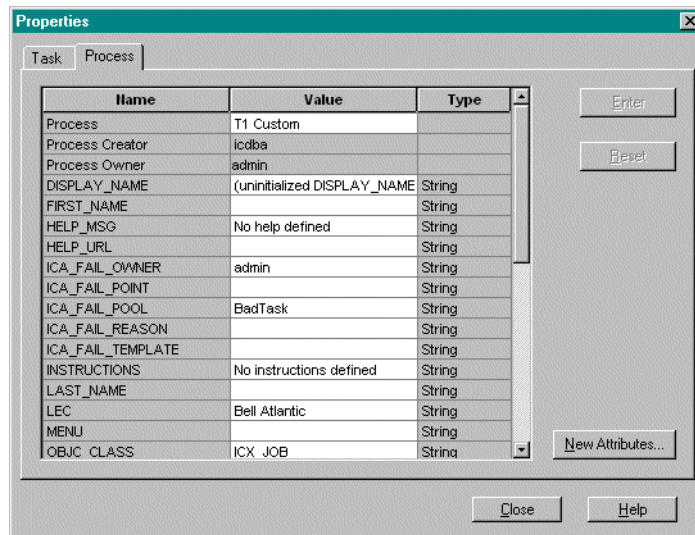
5. Click **OK** or double-click the process name. The Process Query dialog box closes. The process appears in the Design Pad.
6. Repeat Steps 1 through 5 but instead of selecting All eLink Templates in Step 2, select All eLink Palettes.

If no palettes appear in the listbox (Step 2), you must generate one by returning to the Business Interface Window and generate a palette according to the instructions in the Generating a Palette section of this chapter.

Viewing process attributes

In the Property Sheet, the Process tab (shown in Figure 4-14) displays the name, value, and type of each process attribute.

Figure 4-14 Process tab of the Property Sheet



Different attributes take different kinds of values: strings, datetimes, durations, and integers. In most cases, you can select a value from a listbox. In others, you must enter the value yourself, in which case you may only enter the right kind of value. For example, you must enter the task Due Date properly or it will not be accepted.

Summary of built-in process attributes

Table 4-3 describes the internal process attributes. Some internal attributes have meaning only for process instances; the Property Sheet does not show these attributes for process definitions. Moreover, attributes for process instances are read-only in the Property Sheet. For example, the User Working On attribute indicates who is currently working on a task. The Business Process Engine sets this attribute automatically for each task in a process instance; you cannot change it in the Process Designer.

Table 4-3 Internal process attributes

Name	Value	Type
Name	name of process	string
Process Creator	name of creator of process	string
Time Created	time the process was created	datetime

Validating process definitions

The Business Process Window has validation tools that let you view information about tasks in a process in summary format. When defining processes manually, these tools are necessary to ensure consistency with each definition. However, the eLink Process Design Assistant automates key portions of the procedure, thus removing the need to manually validate definitions.

Working with Tasks

In this section, you will learn how to copy and modify tasks in a process definition.

Copying tasks

You can copy a task in the Task pane and paste it anywhere in the process structure.

Note: The task copying most useful with regard to eLink is to copy a task from a palette to a process definition template.

Each copy includes the task and all its descendants, not just its immediate subtasks. When you paste a task, you insert the entire task structure. You cannot copy multiple tasks at the same time.

There are two ways to copy a task:

- Copy and paste
- Drag-and-drop

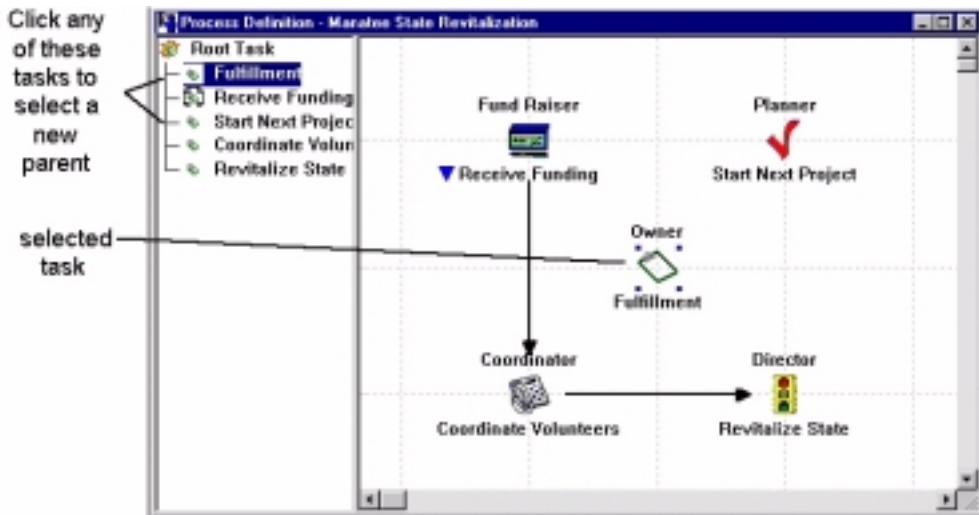
You can use the drag-and-drop method only between tasks displayed simultaneously in the task pane; in other words, sibling tasks. However, the new parent task does not have to be visible in the task pane; when you drag-and-drop a task, the task pane scrolls automatically when you reach the window edge.

Instructions for Copying a Task

To copy a task:

1. Select the task in the Task pane.
2. Choose **Edit > Copy**. Move the cursor into the hierarchy pane and click the task's new parent, as shown in Figure 4-15.


Figure 4-15 Copying a task in the Design Pad



1. Choose **Edit > Paste**. This inserts the task and all its descendants into the process structure, and displays the task icon(s) in the task pane.
2. Adjust the task layout, as needed.

Instructions for Copying a Task with Drag-and-Drop

To copy a task with drag-and-drop:

1. Press and hold your left mouse button over a task in the task pane – *do not release the mouse button*. The cursor changes to a drag icon: 
2. Drag the task to another task in the task pane and drop it on the task icon.
3. A popup menu is displayed from which you can choose the kind of operation you want to perform:
 - Choose **Create Subtask** to insert the task and its descendants into the process structure. Choosing this option inserts the task and its descendants into the process structure.
 - Choose **Cancel** to cancel the operation.

Working with task objects in the Design Pad

In this section, you will learn how to manipulate tasks using the Design Pad interface.

Selecting and moving tasks

The select-and-move layout mode is the default mode; you can select and move a task without choosing a command from the menu bar or toolbar. If you have not yet selected another layout mode, you need only to perform the correct mouse action to select or move a task. If you have already chosen another layout mode, you can use special mouse operations to select or move a task without leaving the mode.

Instructions for Selecting and Deselecting Tasks and Dependencies

To select and deselect tasks and dependencies:



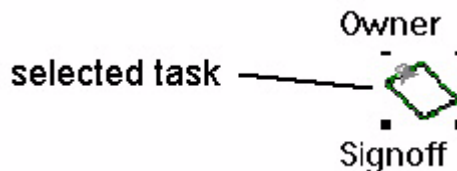
1. Choose **Edit > Select and Move**, or click the  button on the toolbar.
2. Move your mouse over the task pane. The cursor changes to a selection arrow: 
3. Move your mouse over a task icon (or dependency arrow) and click left. Four handles appear at the icon corners, as shown in Figure 4-16.


Figure 4-16 Selecting a task in the task pane



4. To deselect the task or dependency, left-click in the task pane background.


Instructions for Moving a Task

To move a task:

1. Select the task you want to move.
2. Press and hold the left mouse button. The cursor changes to a drag icon: 
3. Drag the task to a new location and release the mouse button.


Instructions for Selecting a Task in Another Layout Mode

To select a task in another layout mode:

1. Press and hold the **Ctrl** key and click right. The cursor changes to a selection arrow: 
2. Left-click the task you want to select. Handles appear at the icon corners indicating that the task is selected. The Design Pad returns to the previous layout mode.

Instructions for Moving a Task in Another Layout Mode

To move a task in another layout mode:

1. Press and hold the **Ctrl** key and click right. The cursor changes to a selection arrow: 
2. Move the cursor over the task icon and press and hold the left mouse button.
3. Drag the task to a new location and release the mouse button. The Design Pad returns to the previous layout mode.

Changing the attribute display

Each task icon in the Design Pad has two display areas:

- The area *above* the task icon can be set to display the value of any process or task attribute. This field is not editable. By default, the Role attribute is displayed. It is suggested that you change the displayed attribute to the PerformCondition.
- The area *below* the icon shows the task name.

Instructions for Changing the Attribute Display

To change the attribute display:


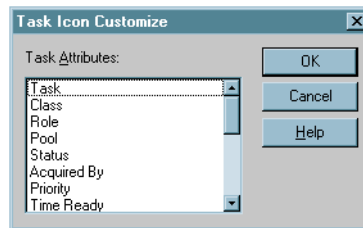
1. Choose **View > Customize**, or click the  button on the toolbar. The Task Icon Customize dialog box appears, shown in Figure 4-17.

Figure 4-17 The Task Icon Customize dialog box



2. In the **Task Attributes** listbox, select the attribute you want to display above each task icon and click **OK**. You can only select one attribute. The dialog box closes and the attribute field above the task icon is updated with the new attribute value.

Naming tasks

You can edit task names when using tasks created manually (for instance, parent tasks into which you are placing eLink-generated subtasks). However, when using eLink, the services that run in the applications connected into the eLink environment depend on the names of the tasks used in the Process Design Assistant Business Interface Window. Therefore, generated tasks must not be renamed in the Design Pad as they will not work properly in the eLink environment.

Aligning tasks

You can automatically place and align tasks in rows or columns in the Design Pad using the Snap to Grid menu option.

When you enable Snap to Grid:

- Existing tasks are not automatically aligned unless you previously aligned them.
- Tasks are aligned when you move them.

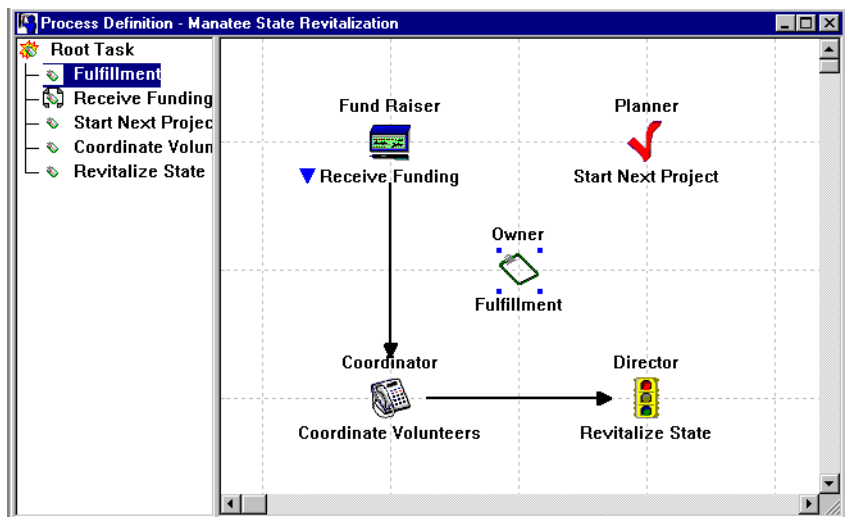
When you close the Process Designer, the state of the Alignment grid is saved to the initialization file and is resumed at the next session.

Instructions for Aligning Tasks

To align tasks:

1. Choose **Edit > Snap to Grid** and **View > Show Grid**. An alignment grid appears as a series of light gray intersecting horizontal and vertical lines, shown in Figure 4-18.

Figure 4-18 Using the alignment grid



Instructions for Hiding the Alignment Grid

To hide the alignment grid:

Choose **View > Hide Grid**.

Instructions for Showing the Alignment Grid

To show the alignment grid:

Choose **View > Show Grid**.

Adding dependencies

Dependencies between tasks determine the order in which tasks can be performed.

The Create Dependencies command lets you add dependencies to tasks. You can also add dependencies while performing another layout operation.

Instructions for Adding a Dependency Between Tasks

To add a dependency between tasks:



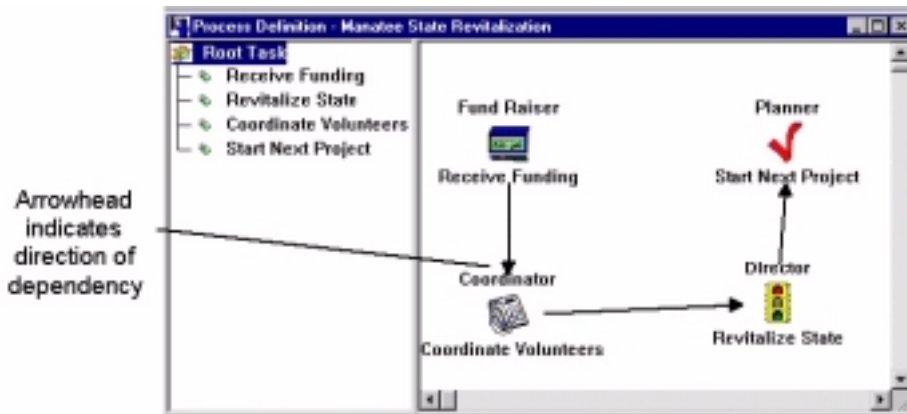

1. Choose **Edit > Create Dependencies**, or click the  button on the toolbar.
2. Move your mouse into the task pane. The cursor changes to a dependency icon: 
3. Press and hold your left mouse button on the task you want to create the dependency from.
4. Drag the cursor to the dependent task and release the mouse button. A dependency arrow appears between the two tasks, as shown in Figure 4-19.

Figure 4-19 Creating a dependency between tasks



Instructions for Adding a Dependency in Another Layout Mode

To add a dependency in another layout mode:



1. Press and hold the **Ctrl** and **Shift** keys and click right. The cursor changes to a dependency icon: 
2. Press and hold your left mouse button on the task you want to create the dependency from.
3. Drag the cursor to the dependent task and release the mouse button. A dependency arrow appears between the two tasks, as shown in Figure 4-19. The Design Pad returns to the previous layout mode.

Deleting tasks and dependencies

You can remove both tasks and dependencies from the process structure.

Instructions for Deleting Tasks and Dependencies

To delete a task or dependency:

1. Choose **Edit > Select and Move**, or click the  button on the toolbar.
2. Move your mouse into the task pane. The cursor changes to a selection arrow: 
3. Select the task icon or dependency arrow you want to delete.
4. Choose **Task > Delete**, or press the **Delete** key. The task icon or dependency arrow disappears from the task pane.

When you delete a task with subtasks, the subtasks are also deleted. When you delete a task with dependencies, the dependencies are also deleted.

Customizing task icons

You can use customized task icons or your own icons, which can be shared with other users.

If you replace an icon, the icon value is cleared and the new icon is displayed.

Adding customized icons to the icon library

You are provided a custom icon library. These icons are located in %IC_HOME%\config\icons. You can use these icons or you can add your own customized icon files to this directory.

If you want to share icons with another user, you and the other user must add the same icon libraries to your icons directory. If you do not have a matching icon library, the default icon appears rather than the customized icon.

The icon files are typically the ICO file type. You can store these icon files in a single location if desired.

Changing task icons

When you create a task using the Design Pad, the task is displayed using the current default icon. That icon is associated with the task and is displayed whenever you open that process on your PC.

Instructions for Showing or Hiding the Task Icon Palette

To show or hide the Task Icon Palette:

Choose **View > Task Icons > Show Palette**. To hide the palette, choose **View > Task Icons > Hide Palette**.

Instructions for Changing the Icon for an Existing Task Using the Task Icon Palette

To change the icon for an existing task using the Task Icon Palette:

1. Select the task whose icon you want to change.
2. Choose **View > Task Icons > Show Palette**. The Task Icon Palette appears, shown in Figure 4-20.

Figure 4-20 The default Task Icon Palette



3. Select the icon you want to use.

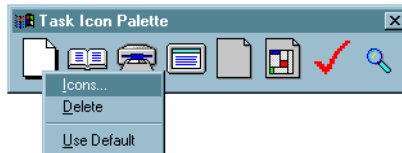
4. Choose **Task > Replace Icon** to replace the existing icon with the current icon from the Task Icon Palette.
5. Repeat until you have changed all the task icons that you want customized.

Instructions for Adding an Icon to the Task Icon Palette

To add an icon to the Task Icon Palette:

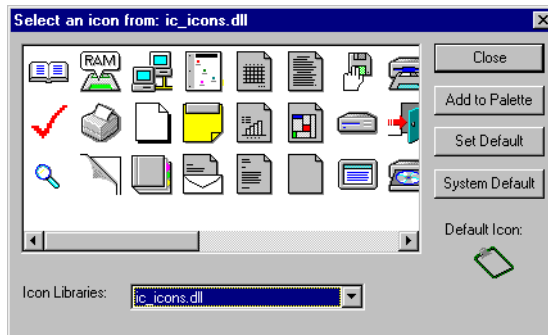
1. In the Task Icon Palette, click right. A pop-up menu appears, shown in Figure 4-21.

Figure 4-21 The pop-up menu in the Task Icon Palette



2. Choose the **Icons** menu item. The Select an Icon dialog box appears, shown in Figure 4-22.

Figure 4-22 The Select an Icon dialog box



3. Select a library from the **Icon Libraries** listbox, which displays icons and lets you browse different libraries.
4. Select an icon and click **Add to Palette**.
5. When you are finished making selections, click **Close**.

Instructions for Removing an Icon from the Task Icon Palette

To remove an icon from the Task Icon Palette:

In the Task Icon Palette, right-click on the icon you want to remove. A pop-up menu appears, shown in Figure 4-21. Choose the **Delete** menu item.

Instructions for Selecting a Default Icon using the Task Icon Palette

To select a default icon using the Task Icon Palette:

In the Task Icon Palette, right-click on the icon you want to become your default icon. A pop-up menu appears, shown in Figure 4-21. Choose the **Set Default** item from the pop-up menu.

Instructions for Setting the Default Icon Using the Select an Icon Dialog Box

To set the default icon using the Select an Icon dialog box:

1. Choose **View > Task Icons > Icons**. The Select an Icon dialog box appears, shown in Figure 4-22.
2. Select the icon you want to become the default.
3. Click **Set Default**. The selected icon is displayed under **Default Icon:** in the lower right-hand corner.

Instructions for Using the System Default Icon as Your Default Icon

To use the system default icon as your default icon:

1. Choose **View > Task Icons > Icons**. The Select an Icon dialog box appears, shown in Figure 4-22.
2. Click **System Default**. The system default icon is displayed under **Default Icon:** in the lower right-hand corner, and becomes your default icon.

Specifying attributes

The Property Sheet lets you view and modify task and process attributes.

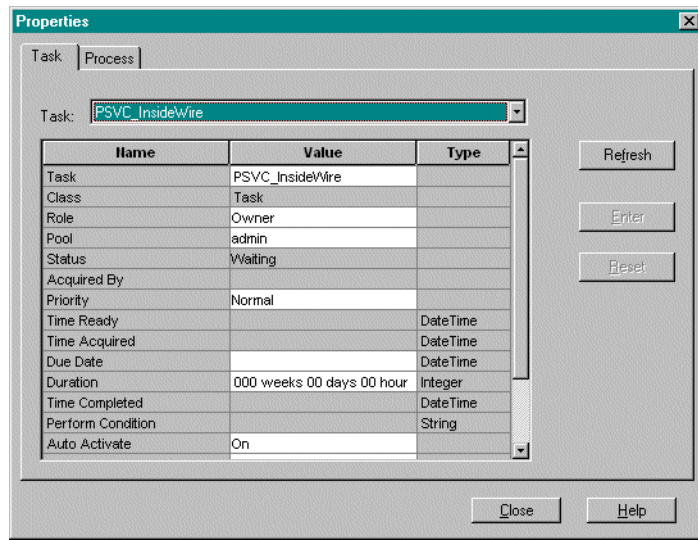
Instructions for Opening the Property Sheet

To open the Property Sheet:

Choose **View > Properties**. The Property Sheet appears, shown in Figure 4-23. The Task tab displays the name, value, and type of each task attribute.

- If a task is selected in the Design Pad, the Task tab shows the attributes for that task.
- If no task is selected in the Design Pad, the Task tab shows the attributes for the root task of the process you are working on.

Figure 4-23 Task page of Property Sheet



Instructions for Viewing the Attribute Values for a Task

To view the attribute values for a task:

There are two methods for displaying the attributes of a particular task in the Property Sheet:

- Select the task in either pane of the Design Pad. This is the recommended method. Selecting the task this way enables the tooltip-style floating display of attribute values.
- Select the task in the **Task** combo box in the Property Sheet. If the task is selected via the combobox, the floating display is not available.

Instructions for Setting an Attribute Value or Modifying an Existing Value

To set an attribute value or modify an existing value:

1. Select the task whose attribute values you want to modify, using any of the methods in Instructions for Viewing the Attribute Values for a Task.
2. In the Property Sheet, select the attribute whose value you want to change. (Use the scroll bars, if necessary, to bring the attribute into view.) The default attribute value (if any) appears in the Value field, next to the attribute name.
3. Select a value from the **Value** listbox. If the attribute does not have a set of predefined values, enter a value in the Value field. Use the correct format for any value you enter manually (see Table 5-3).
4. To reset an incorrect value, click **Reset**. If the attribute had no previous value, the default value is displayed. If the attribute has no default value, the Value field will be empty.

Note: You can only reset values that have not been committed to the database. Once you click **Enter**, you cannot reset the value.

5. To enter the value, click **Enter** or press the **Return** key. The new value appears next to the attribute name.

Instructions for Adding an Attribute to be Used in a Perform Condition

To add a process attribute to be used in a PerformCondition:

1. Select **View --> Properties**.
2. Click on the Process tab of the Properties dialog box.
3. Click on the **New Attributes** button.

Note: You can only perform Step 3 if you have administrative privileges for your processes.

4. Select the data type of the attribute that you want to add.
5. Type the attribute name, then click on OK.

5 Making eLink Processes Work

In previous instructions, you have designed and built a process flow from tasks. Now, you must add the finishing touches that will make your process flows work correctly with eLink. The following topics are covered in Making eLink Processes Work:

- Business Process and Task Parameter Attributes
- Data Flow Through the Business Process Engine
- Working with Conditions
- Working with Process Instances

Business Process and Task Parameter Attributes

The eLink Process Design Assistant assigns attributes for task parameters during palette generation. These attributes are used to implement process flows. This section explains the implementation procedure.

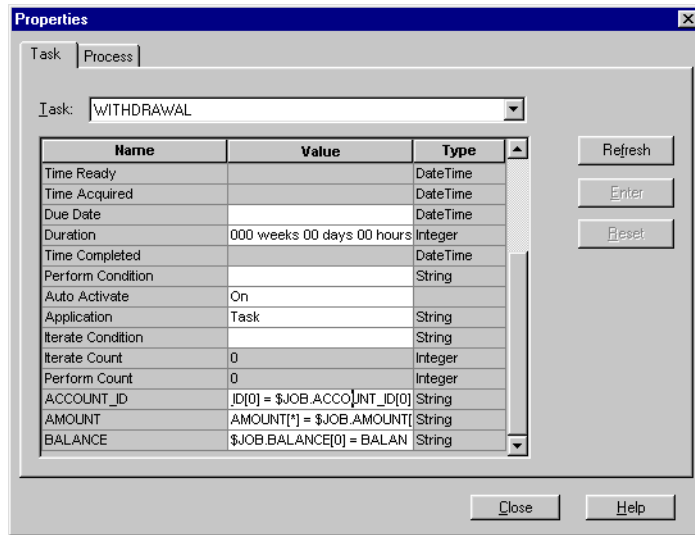
Parameter Attributes & Default Assignments

When palettes are generated, task attributes are generated for every parameter of the included operations. Similarly, template generation creates attributes for the Root Task of the process. These parameter attributes hold the expressions which instruct the eLink Business Process Option's agents as to how to move data to and from the other components (Data Integration Option and application adapters) within the eLink environment. Default assignments are also generated for each parameter.

So, for example, the attributes and default parameter assignments displayed in Figure 5-1 are created for the Withdrawal task during palette generation. The parameter shown is the input parameter AMOUNT. Because it is an input parameter, it is given the assignment `AMOUNT[*] = $JOB.AMOUNT[*]`. The output parameter BALANCE is given the default assignment `$JOB.BALANCE[*] = BALANCE[*]`.

The syntax and meaning of these expressions is explained in the next section.

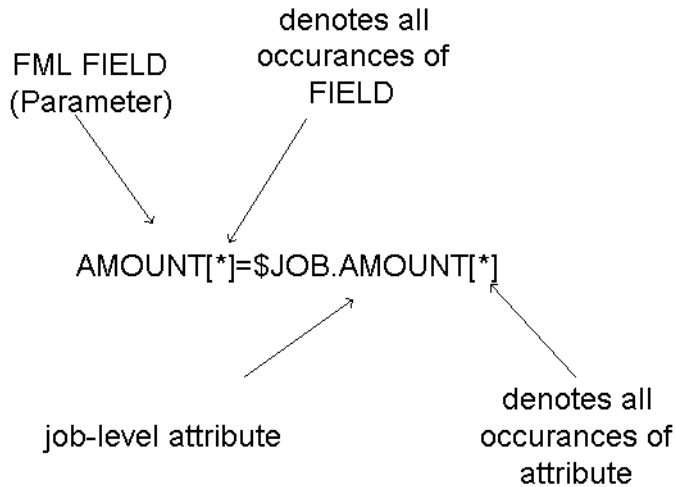
Figure 5-1 Generated Attributes & Default Parameter Assignments



Parameter Assignments Explained

Consider the default parameter assignment for AMOUNT (shown in Figure 5-2).

Figure 5-2 AMOUNT Parameter Assignment (Analyzed)



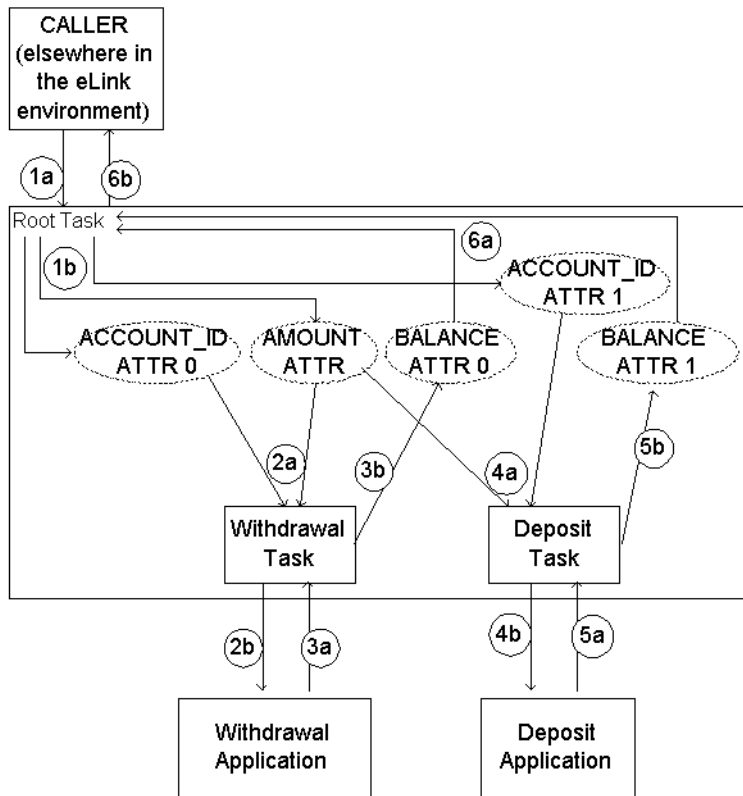
This statement assigns the job-level value of AMOUNT to the FML field AMOUNT. Using the asterisk in brackets `[*]`, this is done for all occurrences of AMOUNT.

Data Flow Through the Business Process Engine

The secret to making processes work is to make sure that the right data is flowing through the right places in the process.

Figure 5-3 shows the data flow through a process.

Figure 5-3 Data Flow Through the Transfer Process



Data Flow By the Numbers

Refer to Figure 5-3 above as the numbered items are explained.

In Step 1a, the caller calls the Transfer process and supplies three input values – the two account numbers (for the withdrawal and deposit accounts, respectively) and the amount to transfer. The input FML Fields from the caller are **ACCOUNT_ID[0]**, **ACCOUNT_ID[1]**, and **AMOUNT[0]**. However, before these values can be used, they must be assigned to job-level attributes.

As Step 1b indicates, these inputs are assigned to job-level attributes according to the expressions stored in the Root Task parameter attributes. All movement of data between FML Fields and attributes within a process is controlled by parameter

assignment expressions. For inputs and outputs to the entire process (at the start and end of the flow), these expressions are stored in the Root Task parameter attributes. The parameter attributes are automatically added to the Root Task by the eLink Process Design Assistant during Template Generation. For this example, we will use the same names for the FML Fields and the corresponding job-level attributes. Since they are job-level attributes they will be qualified by the prefix \$JOB.

Hence, the correct parameter assignment expressions for the Root Task are:

```
$JOB.ACCOUNT_ID[ 0 ]=ACCOUNT_ID[ 0 ]  
$JOB.ACCOUNT_ID[ 1 ]=ACCOUNT_ID[ 1 ]  
$JOB.AMOUNT[ 0 ]=AMOUNT[ 0 ]
```

This indicates that:

- The first instance of the Transfer process' ACCOUNT_ID attribute should be set equal to the value of the first instance (occurrence 0) of the ACCOUNT_ID FML Field.
- The next instance of the ACCOUNT_ID attribute should be set equal to the value of the second instance (occurrence 1) of the ACCOUNT_ID FML Field.
- The first instance of the Transfer process' AMOUNT attribute should be set equal to the value of the first instance (occurrence 0) of the AMOUNT FML Field.

Using a wildcard to refer to all instances of a given attribute or field can further shorten this expression. This is the syntax used by the eLink Process Design Assistant when generating parameter attributes:

```
$JOB.ACCOUNT_ID[ * ]=ACCOUNT_ID[ * ]  
$JOB.AMOUNT[ * ]=AMOUNT[ * ]
```

This indicates that:

- Every instance of the Transfer process' ACCOUNT_ID attribute should be set equal to the value of the corresponding occurrence of the ACCOUNT_ID FML Field.
- Every instance of the AMOUNT attribute should be set equal to the value of the corresponding occurrence of the AMOUNT FML Field.

While we will continue to indicate specific instances of the attributes and FML Field occurrences throughout this example, you will typically work with wildcards.

As the process flow runs, the Withdrawal Task becomes ready to execute. In Step 2, the FML Fields for the Withdrawal operation, ACCOUNT_ID[0] and AMOUNT[0] are assigned values from \$JOB.ACCOUNT_ID[0] and \$JOB.AMOUNT[0] respectively:

```
ACCOUNT_ID[0]=$JOB.ACCOUNT_ID[0];  
AMOUNT[0]=$JOB.AMOUNT[0]
```

We were looking at the FML Fields ACCOUNT_ID[0] and AMOUNT[0] *arriving* into the process in Step 1. The values stored in the \$JOB.ACCOUNT_ID[0] and \$JOB.AMOUNT[0] process attributes during Step 1 will be *sent* (as FML Fields) to the application in Step 2. This explains why the Root and Withdrawal assignments seem like they are reversed in order.

In Step 3, the result of the funds withdrawal is the ending balance for Account 1, so the parameter assignment for the withdrawal result is:

```
$JOB.BALANCE[0]=BALANCE[0]
```

As our process flow continues (not illustrated is the dependency that prevents the Deposit Task from starting before the Withdrawal Task has completed), we arrive at Step 4.

In Step 4, the FML Fields to be sent to the Deposit operation, AMOUNT[0] and ACCOUNT_ID[0] are assigned values from \$JOB.AMOUNT[0] and \$JOB.ACCOUNT_ID[1] respectively. Note that the second instance of the Account Number (the Deposit Account's Number) is being used. However, since the Deposit operation only expects a single Account Number occurrence zero of the FML Field must be sent. This implies the following parameter assignments:

```
AMOUNT[0]=$JOB.AMOUNT[0]  
ACCOUNT_ID[0]=$JOB.ACCOUNT_ID[1]
```

The result of the funds deposit is the ending balance for Account 2, so the parameter assignment for the deposit result is:

```
$JOB.BALANCE[1]=BALANCE[0]
```

Again we can see that the second instance of the Balance process attribute is being updated.

At this point, we have the resulting Balance for the Withdrawal Account in the first instance of the job-level Balance attribute. The second instance of the Balance attribute contains the final Balance for the Deposit Account. Our caller is expecting these results to be returned in two occurrences of the FML Field BALANCE.

Therefore, the parameter assignments on the output side of the process (Step 6) are:

```
BALANCE[ 0 ]=$JOB . BALANCE[ 0 ] ;  
BALANCE[ 1 ]=$JOB . BALANCE[ 1 ]
```

Data Type Management

The data types supported by FML32 and the Business Process Engine must be mapped to support moving data between the two environments. When adding an attribute for use in a PerformCondition, use Table 5-1 to determine the data type of the attribute.

The type of the Process attribute will be either an Integer (signed long integer), String, or DateTime. The eLink agents will convert values in the best possible way. For example, eLink does not support a native DateTime value. These values will be converted to and from FML strings. Similarly, floating point FML fields will be stored as strings. The eLink agents will use the datatype information encoded within the FML Field Ids, together with the attribute type information to determine what conversions to make.

Table 5-1 Data Type Equivalents

FML type	Process Engine type
Char	CxxIcString
String	CxxIcString ¹
CARRAY	Not Supported
String	CxxIcDateTime ²
Short	CxxIcInteger
Long	CxxIcInteger
Float	CxxIcString

Double	CxxIcString
--------	-------------

2. The date-time format must be unambiguously specified to ensure proper encoding and decoding as strings. As with the Predicates used in Perform and Iterate Conditions, the format of a date-time string is 'yyyy-mm-dd hh:mi:ss', where “yyyy” indicates the year, “mm” the month, “dd” the day of month, “hh” the hour in 24-hour notation, “mi” the minute, and “ss” indicates the second. Either the date or time portion may be omitted. If the date is omitted, it is replaced by the current date. If the time is omitted, it is replaced by midnight (all fields 0).

Working with Conditions

You can use conditions to manage the flow in a process. There are two mechanisms for specifying conditions: PerformConditions and IterateConditions.

- A PerformCondition is a task property that determines if the task will be performed or skipped. It is evaluated when a task is about to transition from Waiting to Ready: if True, the task becomes Ready; if False, the task becomes Skipped.
- An IterateCondition is a task property that determines if the task will be either completed or iterated. It is evaluated when the worker requests that the task be completed: if True, the task becomes Ready again; if False, the task becomes Done.

Note: PerformConditions and IterateConditions use the same syntax. We use the term PerformCondition to refer to both, except where explicitly noted. Also, these conditions must be based on attributes defined in the Process Design

Assistant Business Interface Window. Attributes created in the Design Pad will not be recognized by eLink. Therefore, the conditions applied to them will not be recognized, hence not applied.

General form of a PerformCondition or IterateCondition

A PerformCondition has the following general format:

```
attribute_or_property operator value [ logical operator ...]
```

Attribute and property names are case-sensitive, and must be entered exactly as stored in the database. Usually, such names are uppercase. The specified value must have the same datatype as the attribute; for example, integer or string.

You can combine attribute-operator-value statements with logical operators to form complex PerformConditions. The maximum length of a PerformCondition is 254 characters.

Attributes in PerformConditions

A PerformCondition may specify attributes defined for the job or task. To distinguish between the two, you must prefix the attribute name with one of the following strings:

- \$JOB.
- \$TASK.

The prefix is required, even if the attribute is not defined for both the job and the task.

For example, this PerformCondition checks if the value of the current job's AMOUNT_OWED attribute is greater than 10,000:

```
$JOB.AMOUNT_OWED > 10000
```

You can use the Batch Registry to get a list of attributes for a job or task class. Remember that attribute names are case-sensitive, and must be entered exactly as defined for the job or task.

Properties in PerformConditions

In addition to attributes, PerformConditions can also use some properties defined for a job or task. For example, the following PerformCondition will skip a task if its priority is less than 5:

```
$TASK.PRIORITY < 5
```

One important point concerns the `$TASK.$ITERATE_COUNT` and `$TASK.$PERFORM_COUNT` properties: they are incremented after the PerformCondition is evaluated, and before the iterate condition is evaluated. So, to iterate a task three times, you would use the following:

```
$TASK.ITERATE_COUNT < 3
```

This means: “If this task has not been iterated three times already, iterate it.”

Certain restrictions apply to using properties in PerformConditions; see Using internal attributes for more information.

Operators used in PerformConditions

A PerformCondition can include the operators in Table 5-2. Some operators are valid only with certain data types.

Table 5-2 PerformCondition operators

Operator type	Operator	Meaning
Integer and string operators	=	equal to
	>	greater than
	<	less than
	<=	less than or equal to
	>=	greater than or equal to
	<>	not equal
	Logical operators	AND
OR		logical OR
NOT		logical NOT

Data types used in PerformConditions

Attribute data types include string, integer, and DATETIME. String values must be enclosed in single quotes, whether the value is a single word or many words. String values are case-sensitive. They can contain spaces, semi-colons, and double quotes. For example, the following strings are legal:

- 'Kelly Rivera'
- 'Kelly "Rivera"'

- 'Kelly Rivera;'

However, these two strings are not equivalent:

- 'kelly rivera'
- 'Kelly Rivera'

To test DATETIME attributes, you must specify the date and time value as:

```
DATETIME(value)
```

The *value* format is [yy]yy-mm-dd [hh:mm:ss]. For example:

```
$JOB.INVOICE_DUE_DATE > DATETIME(1995-04-21)
```

You can also test DATETIME values against the CURRENT keyword, which evaluates to the current date and time. For example, to determine if an invoice is past due, you might use this PerformCondition:

```
$JOB.INVOICE_DUE_DATE < CURRENT
```

Compound statements

You can combine attribute-operator-value statements with logical operators to form compound statements. Enclosing statements in parentheses is optional; you need parentheses only to change the standard left-to-right order of evaluation (see next section).

For example, to determine if a candidate is eligible for a loan, you might use this PerformCondition:

```
$JOB.LOAN_AMOUNT > 500 AND $JOB.REFERENCES >= 3 OR $JOB.VP_APPROVAL = 'TRUE'
```

Grouping statements with parentheses

You can use parentheses to group attribute-operator-value statements. For example, you could write the PerformCondition in the previous section like this:

```
( $JOB.LOAN_AMOUNT > 500 ) AND  
( $JOB.REFERENCES >= 3 ) OR ( $JOB.VP_APPROVAL = 'TRUE' )
```

You can also use parentheses to change the order in which InConcert evaluates the parts of a PerformCondition. Statements are evaluated in left-to-right order. Using parentheses, you can make sure that related statements are evaluated together.

For example, this PerformCondition evaluates to true if the date of hire is before January 1, 1991 and the person has a Masters or Ph.D. degree.

```
$JOB.HIRE_DATE < DATETIME(1991-01-01) AND  
($JOB.DEGREE = 'MASTERS' OR DEGREE = 'Ph.D.')
```

But this PerformCondition evaluates to true if the person has a Masters degree and was hired before January 1, 1991, *or* the person has a Ph.D.

```
($JOB.HIRE_DATE < DATETIME(1991-01-01) AND  
$JOB.DEGREE = 'MASTERS') OR $JOB.DEGREE = 'Ph.D.'
```

Using internal attributes

You can include an internal attribute in a PerformCondition, such as Due Date or Duration. In most cases, however, you cannot include an internal attribute in a PerformCondition. The reason is that most built-in attributes are really *properties*, not attributes. You cannot include properties in a PerformCondition.

A property is auxiliary information about an object in the database, such as a process or task. Many properties appear as read-only “attributes” in the Property Sheet, such as Process Created By, User Working On, Time Acquired, and Time Completed. Others, such as the Role, Pool, Due Date, Duration, and Priority, are modifiable task properties; AutoActivate is another modifiable task property.

One of the few built-in attributes that you can include in a PerformCondition is Application, because this is a real attribute. To include a built-in attribute in a PerformCondition, you must specify the attribute name as it is stored in the database, not as it appears in the Attributes window. The Application attribute’s real name is IC_APPLICATION. You can use the Batch Registry utility to list other attribute names for different objects in the database; for details, see the *Business Process Option Administration Guide*.

Summary of built-in task attributes

Table 5-3 lists the built-in internal task attributes. Attributes for tasks in process instances are read-only. For example, the Acquired By attribute indicates which worker acquired a task. The Business Process Engine sets this attribute automatically for each task in an active process; it cannot be changed during process design.

Table 5-3 Internal task attributes

Attribute	Value	Type	Comments
Task	name of task		DO NOT CHANGE THE TASK NAMES OF GENERATED TASKS
Class	the class of the task		
Role	role to which task is assigned		DO NOT CHANGE
Pool	pool to which task is assigned		DO NOT CHANGE
Status	status of the task; default is Waiting		
Acquired By	worker who acquired the task		
Priority	priority of task completion		one of Low, Normal, or Urgent
Time Ready	time the task was ready to be acquired		
Time Acquired	time the task was acquired		
Due Date	date the task is or was due	datetime	
Duration	how long it should take to complete the task	integer	estimated task duration

Table 5-3 Internal task attributes

Attribute	Value	Type	Comments
Time Completed	time the task was completed	datetime	
Auto Activate	whether task should automatically become Ready	boolean	default “On” DO NOT CHANGE
Application	custom application to launch when task is opened	string	default “Task”
Perform Condition	test for task to be performed	string	
Iterate Condition	test for iteration to occur	string	
Perform Count	number of times task has been performed	integer	
Iterate Count	number of times iteration has occurred	integer	

The following restrictions apply to the attribute types shown in Table 5-3:

- Strings have a length limit of 254 characters.
- Date input formats depend on your system’s Regional Settings, as specified in the Control Panel.

About attribute values

An attribute is information about a task or process and is displayed in the task window. An attribute might be information the worker uses while performing the task (such as the due date or priority) or information the worker supplies to complete the task (such as total cost).

Attributes can be defined by their scope:

- *Process* attributes have the same value independent of task and can be used to pass information from one task to another. For example, the TOTAL_COST

attribute may be set by a traveling employee in the Travel Report task, and then used by an auditor in an accounting task.

- Task attributes apply only to individual tasks; its value changes from task to task (for example, *Due Date*).

Iterating tasks and subtasks

When you set an IterateCondition and Iterate Count on a parent task, the task's subtasks do *not* automatically inherit that count.

For example, you might expect that setting a parent task's Iterate Count to 3 – meaning the task should iterate 3 times when its IterateCondition is satisfied – would cause each subtask itself to iterate 3 times for every 1 time the parent iterates. This would result in each subtask being performed a total of 9 times. But this is not the case.

Instead, the subtasks' Iterate Counts are reset to zero (0) at the end of each iteration of the parent task, which results in the subtasks being performed only as many times as the parent task itself is performed – in our example, 3 times.

Thus we can see that the Iterate Count when specified on a parent task has the following semantics: Iterating a parent task means to iterate *as a whole* the group consisting of the parent task plus all its subtasks.

Examples of PerformConditions

The following are additional examples of PerformConditions:

```
$JOB.CLAIM_AMOUNT < 1000
```

The value of the CLAIM_AMOUNT attribute is less than 1000.

```
$JOB.SIZE > 5 AND $JOB.ORIGIN = 'New York'
```

The value of the SIZE attribute is greater than 5 and the value of the ORIGIN attribute is New York.

```
($JOB.CLAIM_AMOUNT > 1000) AND ($JOB.CLAIM_AMOUNT < 10000)
```

The value of the CLAIM_AMOUNT attribute is greater than 1000 and less than 10000.

```
($JOB.REPORT_TYPE = 'shipping damage') AND  
($JOB.DATE_FILED > DATETIME(1995-01-15 17:00:00))
```

The value of the REPORT_TYPE attribute is shipping damage and the report was filed after January 15, 1995 at 5:00 PM.

Adding PerformConditions to tasks

Before adding PerformConditions to a task, you must add some key attributes to it. You use the Design Pad to add Perform and IterateConditions to tasks. The Perform Condition and Iterate Condition Wizards consist of pages that assist you as you create a Perform or IterateCondition.

Instructions for Specifying a PerformCondition

To specify a PerformCondition:



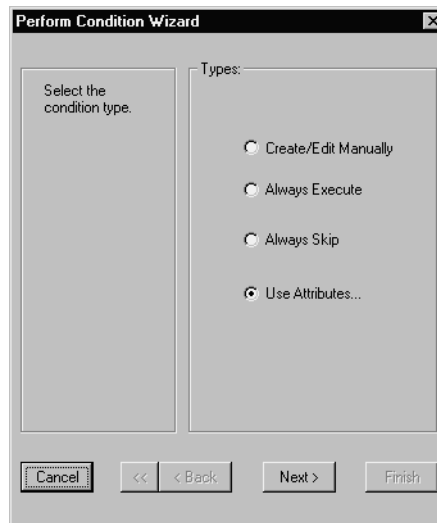
1. Select the task you want to add a PerformCondition to.
2. In the Task tab of the Property Sheet, select the Perform Condition attribute. The  button next to the attribute is enabled.
3. Click the  button. The first page of the Perform Condition Wizard appears, shown in Figure 5-4.

Figure 5-4 First page of the Perform Condition Wizard



4. Select one of the following PerformCondition types and click the appropriate radio button:
 - **Create/Edit Manually.** Click **Next** and go on to “To create a PerformCondition with evaluation expressions:” on page 22.
 - **Always Execute.** Click **Finish**.
 - **Always Skip.** Click **Finish**.
 - **Use Attributes.** Click **Next** and go on to Instructions for Creating a PerformCondition based on Task Attributes.

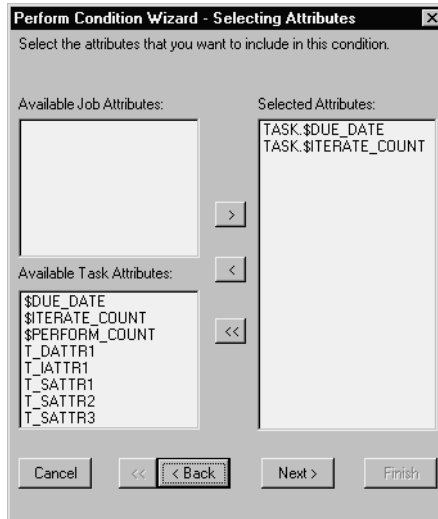
If you want to make changes on an earlier page while you are working in the Perform Condition Wizard, you can return to that page by clicking the **Back** button.

Instructions for Creating a PerformCondition based on Task Attributes

To create a PerformCondition based on task attributes:

1. Click the **Use Attributes** radio button in the first page of the Perform Condition Wizard, then click **Next**. The Selecting Attributes page appears, shown in Figure 5-5.

Figure 5-5 The Selecting Attributes page



2. In the **Available Job Attributes** or **Available Task Attributes** listboxes, select the attribute that you want to include in the PerformCondition you are creating. Click the right arrow button to move it to the **Selected Attributes** box.

Note: eLink features dynamic creation of attributes at runtime. As each task in a process (beginning with the Root Task) executes, parameter assignment expressions are evaluated. An incoming FML field's contents may need to be stored in a process-level (or a parent task's) attribute. If this attribute does not already exist, it will be created by the eLink agent. While this saves a great deal of work, you may find that you need to manually create some of these attributes at design time in order to use them in perform or iterate conditions.

To remove a job or task attribute from the new PerformCondition, select the attribute in the **Selected Attributes** listbox and click the left arrow button to move it back to the **Available Job Attributes** or **Available Task Attributes** listboxes.

To remove all attributes from the new PerformCondition, click the double arrow button.

3. Click **Next**. The Creating Expressions page displays, shown in Figure 5-6. Use this page to create a PerformCondition expression, by selecting a Relation and a Value for each attribute.

Figure 5-6 The Creating Expressions page

Perform Condition Wizard - Creating Expressions

Select a Relation and a Value for each attribute. The valid date format is mm/dd/yy. To change the selected attributes, click the back button to return to the previous page.

Type	Attribute	Relation	Value
Date	TASK.\$DUE_DATE	=	05/06/97
Num	TASK.\$ITERATE_CO	=	2

Relation dropdown options: =, >, <, <=, >=, <>

Buttons: Cancel, <<, < Back, Next >, Finish

4. For each attribute, select an operator from the **Relation** field.
You can change the attribute row order by dragging the button at the left edge of the row up or down the column.
5. Enter a value in the **Value** field. The **Type** field tells you what type of value is required (text, number, or date).
6. Click **Next**. The Combining Expressions page appears, shown in Figure 5-7. Use this page to combine the expressions that you have just created.

Figure 5-7 The Combining Expressions page

Perform Condition Wizard - Combining Expressions

To combine the condition expressions, select AND or OR in the Logical column. To change any of the expressions, click the Back button to return to the previous page.

	Expression	Logical
1	TASK.\$DUE_DATE = 06/06/97	AND
2	TASK.\$ITERATE_COUNT > 2	AND

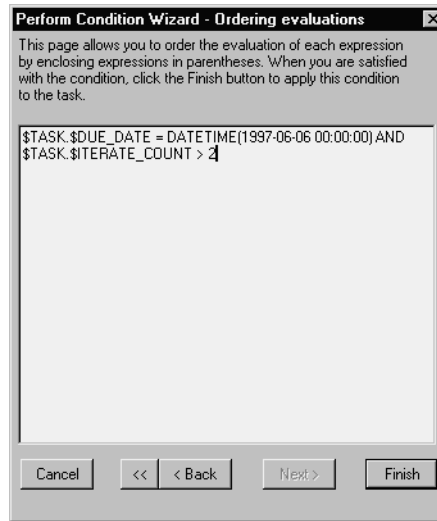
Cancel << < Back Next > Finish

7. Combine the individual expressions using AND/OR operators to construct the type of PerformCondition that you want.

You can change the attribute row order by dragging the button at the left edge of the row up or down the column.

8. Click **Next**. The Ordering Evaluations page appears, shown in Figure 5-8, with the PerformCondition that you have created. You can enclose expressions in parentheses on this page to change the order of the evaluations in the new PerformCondition.

Figure 5-8 The Ordering Evaluations page

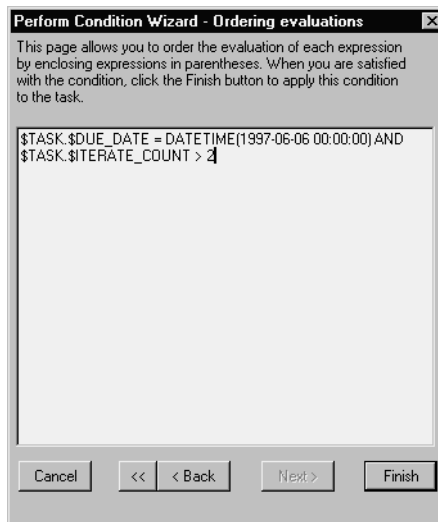


9. Click **Finish**. The Perform Condition Wizard closes and the PerformCondition appears in the Property Sheet next to the attribute name.

Instructions for Creating a PerformCondition with Evaluation Expressions

To create a PerformCondition with evaluation expressions:

1. Click the **Create/Edit Manually** radio button on the first page of the Perform Condition Wizard.
2. Click **Next**. The Ordering Evaluations page appears, shown in Figure 5-9.

Figure 5-9 The Ordering Evaluations page

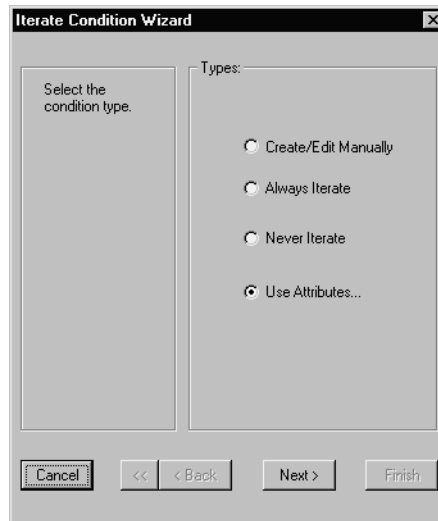
3. In the open area, type in the expression you want to use to evaluate the PerformCondition.
4. When you have finished entering expressions, click **Finish**. The Perform Condition Wizard closes and the PerformCondition appears in the Property Sheet next to the attribute name.

Instructions for Specifying an IterateCondition

To specify an IterateCondition:

1. Select the task you want to add an IterateCondition to.
2. In the Task tab of the Property Sheet, select the IterateCondition attribute. The button next to the attribute is enabled.
3. Click the button. The first page of the Iterate Condition Wizard appears, shown in Figure 5-10.

Figure 5-10 First page of the Iterate Condition Wizard



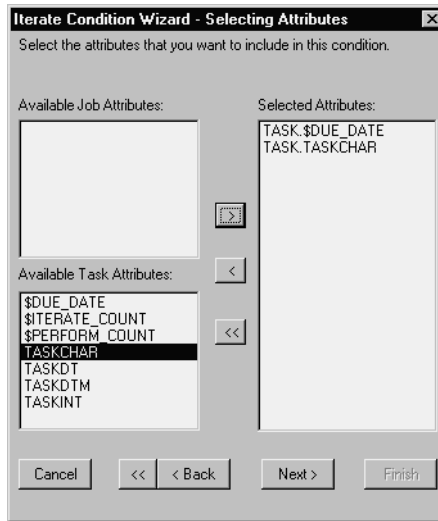
1. Select one of the following IterateCondition types and click the appropriate radio button:
 - **Create/Edit Manually.** Click **Next** and go on to “Instructions for Creating an IterateCondition with Evaluation Expressions” on page 28.
 - **Always Execute.** Click **Finish**.
 - **Always Skip.** Click **Finish**.
 - **Use Attributes.** Click **Next** and go on to To create an IterateCondition based on task attributes:.

If you want to make changes on an earlier page while you are working in the Iterate Condition Wizard, you can return to that page by clicking the **Back** button.

Instructions for Creating an IterateCondition Based on Task Attributes

To create an IterateCondition based on task attributes:

1. Click the **Use Attributes** radio button in the first page of the Iterate Condition Wizard, then click **Next**. The Selecting Attributes page appears, shown in Figure 5-11.

Figure 5-11 The Selecting Attributes page

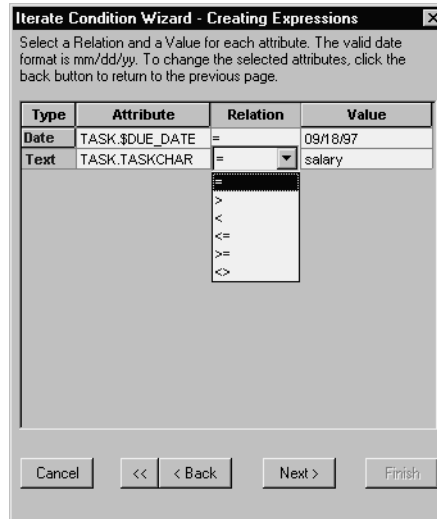
2. In the **Available Job Attributes** or **Available Task Attributes** listboxes, select the attribute that you want to include in the IterateCondition you are creating. Click the right arrow button to move it to the **Selected Attributes** box.

To remove a job or task attribute from the new IterateCondition, select the attribute in the **Selected Attributes** listbox and click the left arrow button to move it back to the **Available Job Attributes** or **Available Task Attributes** listboxes.

To remove all attributes from the new IterateCondition, click the double arrow button.

3. Click **Next**. The Creating Expressions page displays, shown in Figure 5-12. Use this page to create an IterateCondition expression, by selecting a Relation and a Value for each attribute.

Figure 5-12 The Creating Expressions page

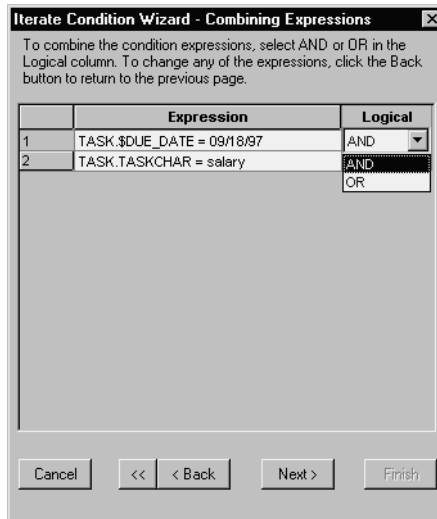


4. For each attribute, select an operator from the **Relation** field.

You can change the attribute row order by dragging the button at the left edge of the row up or down the column.

5. Enter a value in the **Value** field. The **Type** field tells you what type of value is required (text, number, or date).
6. Click **Next**. The Combining Expressions page appears, shown in Figure 5-13. Use this page to combine the expressions that you have just created.

Figure 5-13 The Combining Expressions page

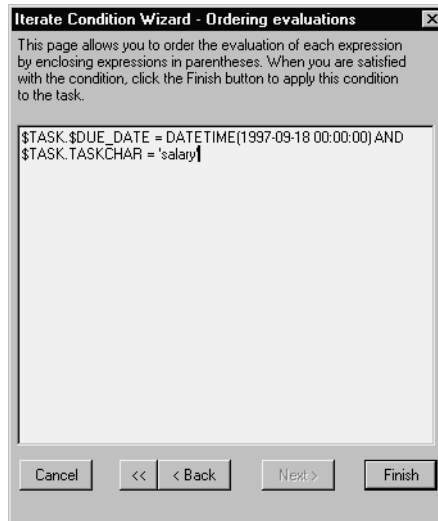


- Combine the individual expressions using AND/OR operators to construct the type of IterateCondition that you want.

You can change the attribute row order by dragging the button at the left edge of the row up or down the column.

- Click **Next**. The Ordering Evaluations page appears, shown in Figure 5-14, with the IterateCondition that you have created. You can enclose expressions in parentheses on this page to change the order of the evaluations in the new IterateCondition.

Figure 5-14 The Ordering Evaluations page



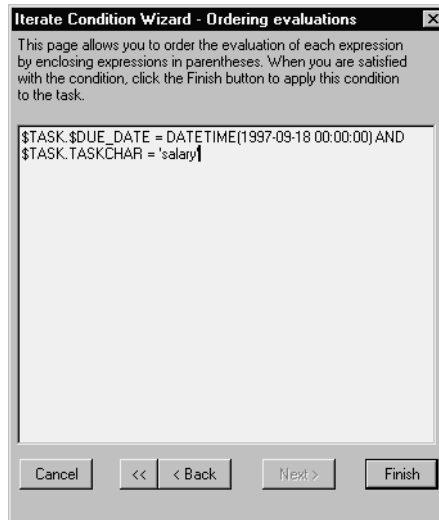
9. Click **Finish**. The Iterate Condition Wizard closes and the IterateCondition appears in the Property Sheet next to the attribute name.

Instructions for Creating an IterateCondition with Evaluation Expressions

To create an IterateCondition with evaluation expressions:

1. Click the **Create/Edit Manually** radio button on the first page of the Perform Condition Wizard.
2. Click **Next**. The Ordering Evaluations page appears, shown in Figure 5-15.

Figure 5-15 The Ordering Evaluations page



3. In the open area, type in the expression you want to use to evaluate the IterateCondition.
4. When you have finished entering expressions, click **Finish**. The Iterate Condition Wizard closes and the IterateCondition appears in the Property Sheet next to the attribute name.

Working with Process Instances

The Process Manager tool lets you view the status of processes in progress. The Business Process Engine also allows you to start processes within this tool. However, since agents are responsible for starting all processes in the eLink environment, you will never need to manually start processes with the Process Manager. In fact, eLink will not successfully run processes started manually.

Instructions for Starting a Process

Processes started manually with the Process Manager will not run correctly in eLink. Processes can be correctly started in the following ways:

- From a web page or other client connected to eLink
- From an adapter connected to eLink that supports requests to eLink
- From the Operation Test utility in the Business Interface Window. The operation tested must be an operation that is implemented as the process flow that you want to run. For instructions on testing an operation, see Test Operation Instructions in Chapter 3, “Specifying Business Service Contracts.”

Managing Processes with the Process Manager

You have privileges to manage any of the processes displayed in the Current Processes folder in the Process Manager window.

You can perform the following procedures with the Process Manager:

- Check and grant privileges on processes, tasks
- Complete processes
- Delete completed processes
- Modify task assignments in process instances
- Suspend and resume processes
- Transfer tasks from one worker to another within processes
- View the status of process instances

Opening the Process Manager

To use the Process Manager, you must have already started the Business Process Window by clicking on the Edit Flow button in the Business Interface Window.

Instructions for Starting the Process Manager

To start the Process Manager:


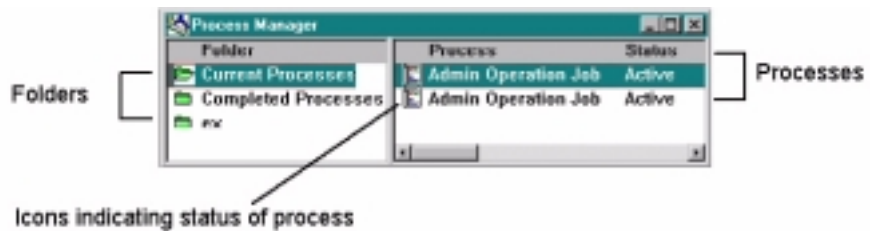
In the main window, choose **File > Open > Process Manager**, or click the  button on the toolbar. The Process Manager window appears with two panes. The pane on the left displays your folders. The pane on the right displays the contents of the open folder. An example is shown in Figure 5-16.

Figure 5-16 The Process Manager window



The icons to the left of each process name represent the status of the process. When you move your cursor over the icon, the status bar shows you the status message.

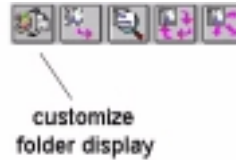
The Process Manager includes two standard folders:

- Current Processes contains all processes that you have started.
- Completed Processes contains all completed processes that you started.

You can also create personal folders to group processes. In the example above, the user has created a personal folder and copied process definitions that she regularly starts.

In addition to the main Process Manager window, the buttons shown in Figure 5-17 appear on the main toolbar. Because eLink automates the starting of processes, only the customize button is useful.

Figure 5-17 Process Manager toolbar buttons



Completing a process

A process automatically completes when the last task in the process is done. You can also force completion of a process.

When you force process completion, tasks with either a Waiting or Ready status are skipped. Tasks that are Acquired are removed from the worker's list of tasks. Changes to documents are discarded, unless the worker checked in the document before you forced the process to completion.

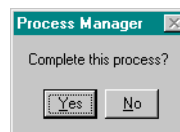
You cannot force a process to completion if another user has acquired the root task. If you acquired the root task, or the root task is Activated or Waiting, you can still complete the process.

Instructions for Forcing a Process to Completion

To force a process to completion:

1. In your Current Processes folder, select the process you want to complete.
2. Choose **Process > Complete**. The Complete Process dialog box appears, shown in Figure 5-18.

Figure 5-18 The Complete Process dialog box



3. Click **OK**.

4. When the process has been successfully completed, another message box appears. Click **OK** to close the dialog box and place the process in your Completed Processes folder.

Deleting a complete process

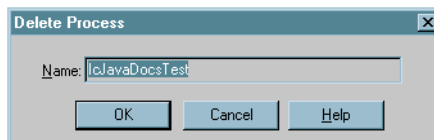
When a process is completed, it is moved into your Completed Processes folder. The process remains in that folder until you delete it.

Instructions for Deleting a Completed Process

To delete a completed process:

1. Open your Completed Processes folder.
2. Select the completed process you want to delete.
3. Choose **Process > Delete**, or press the **Delete** key. The Delete Process dialog box appears, shown in Figure 5-19.

Figure 5-19 The Delete Process dialog box



4. Click **OK**. The process is deleted from your Completed Processes folder and any personal folder in which it is also stored.
5. Click **Cancel** to cancel deleting the process.

Modifying a process instance in progress

You can modify assignments for tasks that have not yet been completed. These assignments include:

- Roles to tasks and pools to roles

- Values of attributes for the process or tasks
- Notes to tasks

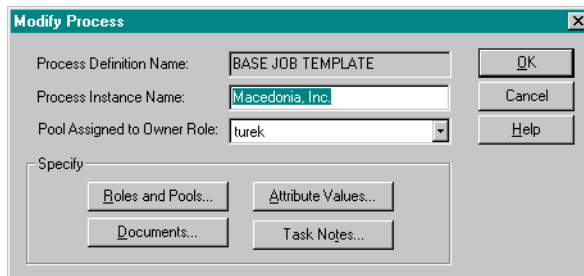
When you modify a process, that process automatically suspends. When you are done modifying the process, it automatically resumes.

Instructions for Modifying Parts of a Process

To modify parts of a process:

1. Select the process you want to modify in your Current Processes folder.
2. Choose **Process > Modify**. The Modify Process dialog box appears, shown in Figure 5-20.

Figure 5-20 The Modify Process dialog box



3. When you are done modifying the process, click **Close**.

Suspending and resuming a process

You can suspend and resume an active process. (You cannot suspend a completed process.) When you suspend a process, the status of all Ready tasks changes to Waiting.

- When you suspend a process, all Ready tasks are removed from each worker's list of tasks. All Acquired tasks remain Acquired and workers can continue to work on those tasks.

- When you resume a process, tasks that had a status of Ready before you suspended the process return to a status of Ready, unless you modified the task to not be Ready.

Instructions for Suspending a Process

To suspend a process:

Select the process you want to suspend in your Current Processes folder. Choose **Process > Suspend**. The icon for the process indicates the process is suspended.

Instructions for Resuming a Process

To resume a process:

Select the suspended process in your Current Processes folder. Choose **Process > Resume**. The icon for the process indicates that the process is active.

Viewing the status of a process

You can view the status of any process in your Process Manager window.

Viewing the status of a process displays a window with two panes. The left pane is the hierarchy pane, which displays the structure of the process. The right pane is the task pane; it displays the flow of the process at a single level in the process structure. For each task in the process, the task pane displays the task name and the status of the task.

You can display other information about the tasks in the task pane, for example, the time completed. You can also view details about individual tasks in the process, which include the following information:

- Task status
- Role and pool assignments
- Name of the worker who acquired the task (if any)
- Time task was completed if the task is done
- Date and Time task is due if a date was assigned

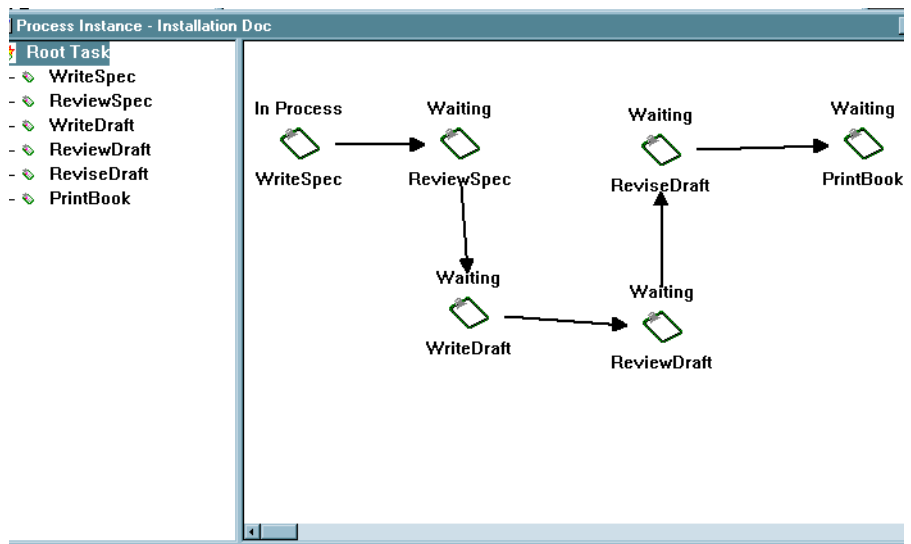
The Process Status window adds a special button to the toolbar, which allows you to change the information displayed above each task.

Instructions for Viewing the Status of a Process

To view the status of a process:

Select a process in the Process Manager window. Choose **Process > View Status** or double-click the process name. The Process Status window appears, shown in Figure 5-21.

Figure 5-21 The Process Status window



Instructions for Viewing Other Levels of the Process

To view other levels of the process:

Select a task in the left pane. Do one of the following:

- Double-click on the task in the left (hierarchy) pane.
- Choose **View > Expand** or **View > Collapse**.

Instructions for Changing the Information Displayed for Each Task

To change the information displayed for each task:


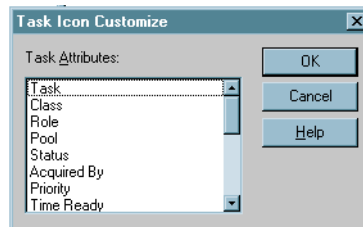
Choose **View > Customize**, or click the  button on the toolbar. The Task Icon Customize dialog box appears, shown in Figure 5-22. Select the attribute you want to display in the **Task Attributes** listbox and click **OK**.

Figure 5-22 The Task Icon Customize dialog box

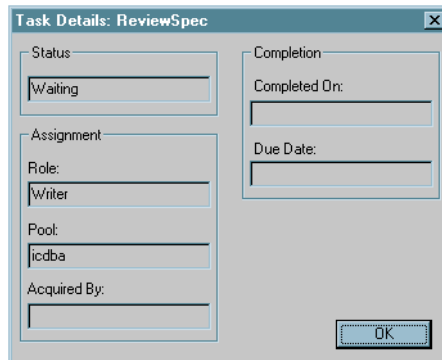


Instructions for Viewing the Details of a Specific Task

To view the details of a specific task:

Select a task in the hierarchy pane. Choose **Task > Show Details**. The Task Details window appears, shown in Figure 5-23. When you are done viewing the information, click **OK** to dismiss the dialog box.

Figure 5-23 The Task Details dialog box

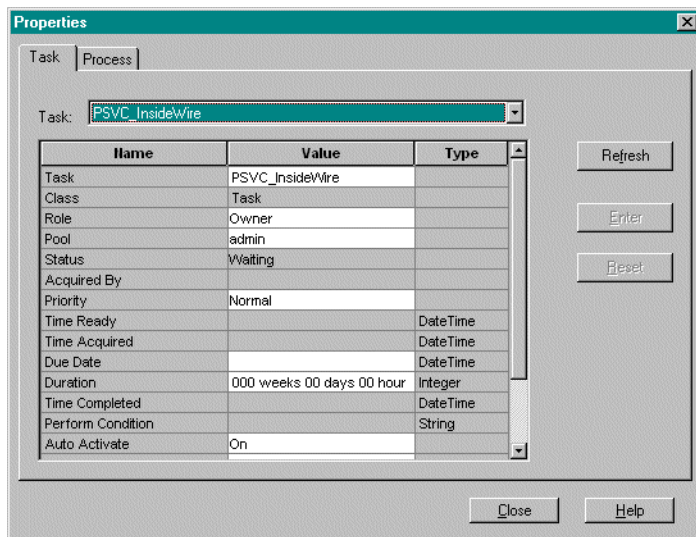


Instructions for Viewing the Properties of a Task

To view the properties of a task:

Select a task in the right pane of the Process Manager window. Choose **View > Properties**. The Task Property Sheet appears in read-only mode, shown in Figure 5-24, allowing you to view but not change the task properties. Click **Close** to close the Property Sheet.

Figure 5-24 The Task Property Sheet



Using folders to organize the Process Manager

The Process Manager lets you organize your work environment: you can use folders to organize your processes, and you can customize how your processes are displayed in a folder.

The Process Manager window includes two standard folders: Current Processes and Completed Processes. To help organize your work in the Process Manager, you can:

- Create personal folders

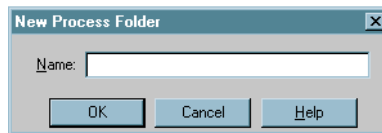
- Rename your personal folders
- Find and add a process definition to a folder
- Move processes between folders
- Delete your personal folders
- Delete individual processes from folders
- Creating personal folders

Instructions for Creating a New Folder

To create a new folder:

1. Choose **Folder > New**. The New Process Folder dialog box appears, shown in Figure 5-25.

Figure 5-25 The New Process Folder dialog box



2. Enter the name for a folder and click **OK**. The new folder is created and displayed in the hierarchy pane of the Process Manager window. The new folder is empty and you can find and add processes to the folder.
3. Click **Cancel** to cancel creating the folder.

Finding and adding process definitions to folders

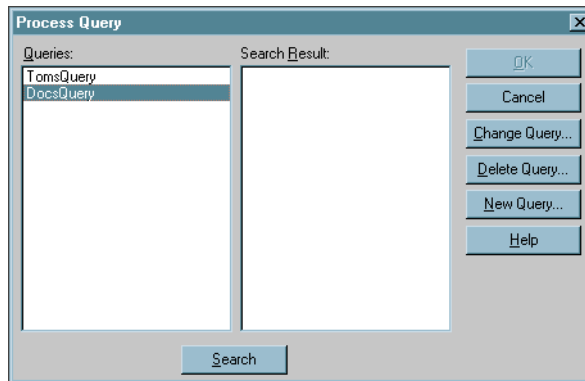
You can keep copies of process definitions in Process Manager folders. You can select these process definitions to start new processes, rather than searching for process definitions each time you need to start a process.

Instructions for Finding a Process Definition and Adding It to a Folder

To find a process definition and add it to a folder:

1. Open the folder to which you want to add a process definition.
2. Choose **Folder > Find and Add Process Definition**. The Process Query dialog box appears, shown in Figure 5-26.

Figure 5-26 The Process Query dialog box



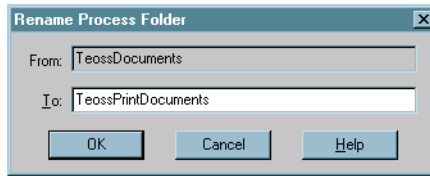
3. In the **Queries** listbox, select the query you want to run.
4. Click **Search**, or double-click the query name. A list of process definitions appears in the **Search Results** listbox.
5. Select one or more process definitions from the **Search Results** listbox.
6. Click **OK**. If you selected just one process definition, you can double-click the process name instead. The process definitions appear in the open folder.

Instructions for Renaming a Folder

To rename a folder:

Select the folder you want to rename. Choose **Folder > Rename**. The Rename Folder dialog box appears, shown in Figure 5-27. Enter a new name for your folder in the **To** box and click **OK**.

Figure 5-27 The Rename Process Folder dialog box



Moving processes between folders

You can move processes between folders using either of these methods:

- The Edit menu
- Drag-and-drop

With the Edit menu, you can copy or move processes. With drag-and-drop, you can only copy processes.

You can copy and move processes freely between personal folders. You can also copy processes in the Current Processes folder and Completed Processes folder. However, you cannot change the contents of the Current Processes folder by moving a process into or out of the folder. Nor can you move a process into the Completed Processes folder.

Instructions for Copying or Moving a Process with the Edit Menu

To copy or move a process with the Edit menu:

Select the process you want to move. Choose **Edit > Cut** or **Edit > Copy**. Open the folder to which you want to move the process. Choose **Edit > Paste**.

Instructions for Copying a Process with Drag-and-Drop

To copy a process with drag-and-drop:

Open the folder containing the process you want to copy. Move the cursor over the process name and press and hold the left mouse button. The cursor changes to a drag icon. Drag the process to the folder in which you want to put the copy and release the mouse button.

Instructions for Deleting a Personal Folder

To delete a personal folder:

Select the folder you want to delete. Choose **Folder > Delete**.

Deleting processes from personal folders

You can remove a process definition from a personal folder, or you can delete a process definition from the system. You must have the necessary privileges to delete a process definition from the system. (If you created the process definition, you automatically have the privileges necessary to delete it.)

Instructions for Removing a Process Definition from a Personal Folder

To remove a process definition from a personal folder:

Open the folder containing the process definition you want to remove and select the process definition. Choose **Edit > Cut**.

Instructions for Removing a Process Definition from a Personal Folder and Deleting It from the System

To remove a process definition from a personal folder and delete it from the system:

Open the folder containing the process definition and select the process definition. Choose **Process > Delete**, or press the **Delete** key.

Customizing how processes are displayed in folders

You can customize how processes are displayed in Process Manager folders by specifying the order in which the processes are displayed and what information is displayed about them. You can sort processes using different options, such as the Process owner or the Time completed. When processes are sorted, they are displayed in groups.

You can also specify the information that is displayed about each process, such as the process definition name or status.

Instructions for Organizing Your Process Display

To organize your process display:


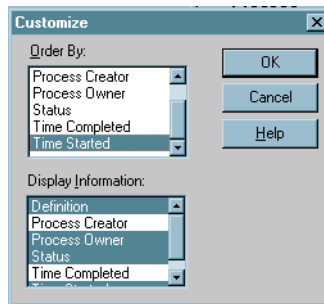
1. Choose **View > Customize**, or click the  button on the toolbar. The Customize Folder Display dialog box appears, shown in Figure .

Figure 5-28 The Customize Folder Display dialog box



2. To specify how the processes should be sorted, select one of the items in the **Order By** listbox.
3. To specify what information is displayed with a process, select one or more items in the **Display Information** listbox.
4. Click **OK**. The right pane of the Process Manager window redisplay the list of processes based on what you selected in the Customize dialog box.

A Interface File (*.IFCE) Reference

This chapter is provided to assist you in using interface files. Interface files are loaded into the Main Window of the Process Design Assistant as an alternative method for creating an interface. The other method of creating an interface is to interactively create one using the tools in the Main Window. The following sections describe how to use interface files:

- Overview
- Loading ATMI Services
- Interface File Example

Overview

eLink provides you a means to integrate enterprise applications. An important part of the integration process is the creation of an interface. In eLink, an interface is composed of a set of operations to be performed on the application to which you are interfacing and their associated inputs and outputs. The definition for this interface can be recorded in a text (ASCII) file, and that file can be loaded into the Process Design Assistant Main Window.

Loading ATMI Services

This section covers the technical details of what occurs when you use the Load button.

Introduction to the Interface Loader

You may have an environment with many ATMI services. Manually creating these definitions using the Process Design Assistant may not be desirable.

The interface loader facility is a part of the Process Design Assistant. This reads a formatted text file consisting of ATMI service definitions and loads them into the Contract Repository Database.

Note: ATMI services are known as *operations* in the Process Design Assistant.

After the Loader has populated the Contract Repository, you may edit services, create new services, and group services using the Process Design Assistant.

See Usage in Chapter 3, “Specifying Business Service Contracts,” for information about using the Process Design Assistant.

About the Interface File

The interface file is a text file that defines services and their associated parameters. The interface loader loads the services defined in the interface file into the Contract Repository using the interface file name. If a load has been performed, the interface exists in the Contract Repository.

Syntax of the Interface Files

Each service definition consists of properties and parameters that have a set number of parameter properties. Each property is represented by a keyword and a value.

Keywords are divided into two levels:

- Service-level

- Parameter-level

Guidelines for Using Keywords

While using the keywords, follow the guidelines in Table A-1.

Table A-1 Guidelines for Using Keywords

Guideline	Example
Each keyword must be followed by an equal sign (=) and the value.	Correct: type=string Incorrect: type
Only one keyword is allowed on each line.	Correct: type=string Incorrect: type=string access=out
Any lines not having an equal sign (=) are ignored.	Correct: type=string Incorrect: type string
Certain keywords only accept a well defined set of values.	The keyword, access accepts these values: in, out, inout, noaccess
The input file may contain multiple service definitions.	service=INQUIRY <service keywords and values> service=DEPOSIT <service keywords and values> service=WITHDRAWAL <service keywords and values> service=TRANSFER <service keywords and values>
Each service definition consists of multiple keywords and values.	service=deposit export=true inbuf=FML32 outbuf=FML32

Keyword Order in the Interface Loader Data File

Keyword order must be maintained within the data files to ensure an error-free transfer during the interface load.

The first keyword definition in the interface loader data text file must be the initial `service=<NAME>` keyword definition (shown in Listing A-1). Following the `service=<NAME>` keyword, all of the remaining service keywords that apply to the named service must be specified before the first `param=<NAME>` definition. These remaining service keywords can be in any order. Refer to Table A-2 for a list of the service keywords and values.

Next, specify all the parameters associated with the service. Following each of the `param=<NAME>` keywords are all the parameter keywords that apply to the named parameter until the next occurrence of a parameter definition. These remaining parameter keywords can be in any order. When all the parameters associated with the first service are defined, specify a new `service=<NAME>` keyword definition.

Listing A-1 Correct Example of Hierarchical Order in a Data File

```
service=<NAME>
<service keyword>=<value>
<service keyword>=<value>
<service keyword>=<value>
param=<NAME>
<parameter keyword>=<value>
<parameter keyword>=<value>
param=<NAME>
<parameter keyword>=<value>
<parameter keyword>=<value>
```

Using Service-Level Keywords and Values

A service is equivalent to an operation. A service definition must begin with the “service=” keyword. See Usage in Chapter 3, “Specifying Business Service Contracts,” for additional information about operations.

To review the service-level keywords and values, see Table A-2.

Table A-2 Service-Level Keywords and Values

Keyword	Value
service	Any ATMI service name
export	true or false (default is false)

Table A-2 Service-Level Keywords and Values

Keyword	Value
inbuf/outbuf	Select one of these buffer types: FML32 NONE

Using Parameter-Level Keywords and Values

A parameter begins with the “param=” keyword followed by a number of parameter keywords until another “param” or “service” keyword, or end-of-file is encountered. The parameters can be in any order after the “param” keyword.

See Usage in Chapter 3, “Specifying Business Service Contracts,” for more information about parameters.

To review the parameter-level keywords and values, see Table A-3.

Table A-3 Parameter-Level Keywords and Values

Keyword	Values
param	Any parameter name
type	byte short integer float double string carray
access	in out inout noaccess
count	Maximum number of occurrences (default is 0). The value for unlimited occurrences is 0. Used only by the Process Design Assistant to format test screens.

Troubleshooting

If you encounter any problems using the interface loader utility, see Table A-4.

Table A-4 Loader Troubleshooting Table

If . . .	Then . . .
the data file is not found	check to ensure that the path is correct Note: UNC path names (of the form \\<computer name>\path) are not directly supported. To load interface files from a remote drive, first map the drive to a local drive letter.
the keyword is invalid	check to ensure that the keyword is valid for the package, service, or parameter
the value of the keyword is null	type a value for the keyword
the value is invalid	check to ensure that the value of a parameter is within the allocated range
the data type is invalid	check to ensure that the parameter is using a valid data type

Interface File Example

The following is an example of an interface file. It can be used to load the interface for the simple funds transfer example that is used throughout this guide.

```
service=SubtractInteger
export=true
inbuf=FML32
outbuf=FML32
param=INTEGER_OPERAND
type=long
access=in
```



```
count=3
param=INTEGER_RESULT
type=long
access=out
count=2

service=AddInteger
export=true
inbuf=FML32
outbuf=FML32
param=INTEGER_OPERAND
type=long
access=in
count=3
param=INTEGER_RESULT
type=long
access=out
count=2
```

Interface File Explained

Let us examine the first few lines of the sample interface file and explain each one as we go.

```
service=SubtractInteger
```

This line specifies an operation in the Business Interface Window called `SubtractInteger`.

```
export=true
```

If this line is set to true, the `SubtractInteger` service is exported. Therefore, it can be tested.

```
inbuf=FML32
outbuf=FML32
```

These two lines set the input and output buffer types to `FML32`, the only type allowed in eLink.

```
param=INTEGER_OPERAND
```

This line makes `INTEGER_OPERAND` a parameter of the `SubtractInteger` operation.

```
type=long
```

A *Interface File (*.IFCE) Reference*

This line specifies that element INTEGER_OPERAND will be of the long integer type. An element can be specified as any of the standard C data types. int=integer variable, long=long integer, float=floating point variable, string=string variable.

```
access=in
```

This line designates parameter INTEGER_OPERAND as an input. The choices for the access setting are in=input, out=output, inout=input/output.

```
count=3
```

This line sets the Occurance variable for INTEGER_OPERAND to "3."

B MATHAPP Tutorial & Reference

MATHAPP is a simple mathematics application package that is written as ATMI services and doesn't require that an external database be installed. This appendix also provides you with a MATHAPP reference. The following sections provide you with a tutorial that uses MATHAPP to form a simplified funds transfer process:

- Creating a Simplified Funds Transfer Process Flow: a Tutorial
- MATHAPP Service Reference

Creating a Simplified Funds Transfer Process Flow: a Tutorial

The following is an example of how a process flow is created. It follows from the example presented in Chapter 1, "eLink Overview," and is carried through the remainder of this user's guide. This example is simplified to allow the use of MATHSERV, a simple math application that is provided with the eLink Business Process Option software. A working process flow can be created according to these instructions using only the installed software.

Logging On

In this series of steps, you log on to the Process Design Assistant.

1. Start the eLink Process Design Assistant. For instructions, see Starting the Process Design Assistant in Chapter 3, “Specifying Business Service Contracts.”

The BEA eLink Business Process Option opening window appears.

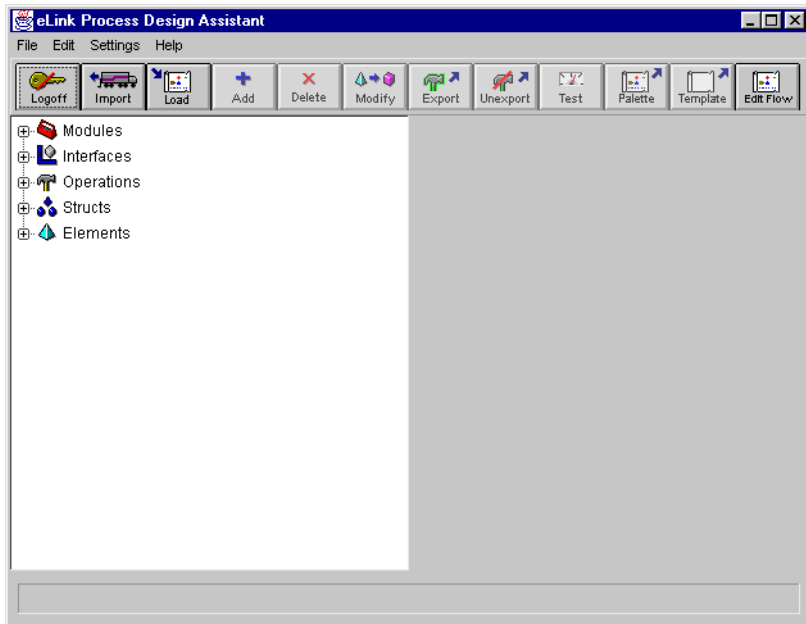
Figure B-1 Process Design Assistant Opening Window



2. Log on to the Process Design Assistant. For instructions, see Logging On to the Process Design Assistant in Chapter 3, “Specifying Business Service Contracts.” Remember to set your logon option defaults according to these instructions before logging on for the first time.

The Process Design Assistant Business Interface Window appears.

Figure B-2 Business Interface Window



Loading an Interface & Adding an Operation

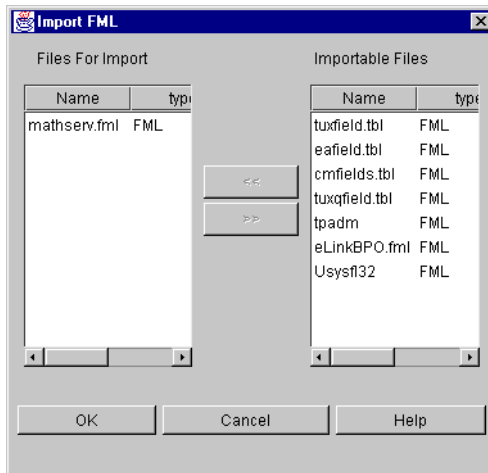
In this series of steps, you set up the interface to the applications connected to eLink. The interface consists of a number of contracts. Contracts are agreements on inputs and expected outputs to and from eLink and the applications connected to eLink. They are also referred to as operations. You also add an operation that will become the template for your process flow.

3. Import `mathserv.fml` (using the Import button). This file contains the FML field definitions needed for MATHSERV. You must import an FML file for particular FML field definitions, but you should do so only once to avoid overwriting subsequent changes to the operation parameters (elements). For instructions, see *Importing FML Files* in Chapter 3, “Specifying Business Service Contracts.”

If `mathserv.fml` does not appear on the list of Importable Files in the Import FML window, see the README file for configuration and installation of the

MATHSERV application. This README file can be found on your server in \$TUXDIR/apps/mathapp.

Figure B-3 Import FML Window



The imported FML fields will appear as elements in the repository tree after refreshing the screen. These elements are now available for use in creating operations and interfaces. Operations & interfaces can be created interactively. However in this tutorial, we will load the interface used by this tutorial.

4. Load SIMPLOPS.IFCE, the SimpleOps interface, using the Load button. You will find SIMPLOPS.IFCE in the Sample directory which is in the directory where you installed the Process Design Assistant client. For instructions, see Loading Externally Described Interfaces in Chapter 3, “Specifying Business Service Contracts.” For an explanation of SIMPLOPS.IFCE, how it works, and its implications on the rest of the eLink architecture, see Appendix A, “Interface File (*.IFCE) Reference.”

The Interface Load Results dialog box appears and lists the actions taken to load the SIMPLOPS Interface.

Note: Because loading an interface may reset certain operational parameters, it should only be done once for a particular interface.

As an alternative, you could interactively create an interface with the SUBTRACT_INTEGER and ADD_INTEGER operations in it. For instructions

in interactively creating an interface, see the Usage section of Chapter 3, “Specifying Business Service Contracts.”

5. Wait until load is completed, then click OK.

The SIMPLOPS interface will appear in the Contract Repository tree after refreshing it (File...Refresh).

6. Add an operation according to the instructions in Adding Operations in Chapter 3, “Specifying Business Service Contracts.” Call it SimpleTeller. Be sure to enter an ATMI name of SimpleTeller. Include the INTEGER_OPERAND and INTEGER_RESULTS elements in it. Modify the SimpleTeller parameter INTEGER_RESULT by changing its Passing Mode to “out.” For instructions in modifying elements, see Modifying Parameters in Chapter 3, “Specifying Business Service Contracts.”
7. Make sure that the Occurrence attribute of INTEGER_OPERAND is set to “3” and the Occurrence attribute of INTEGER_RESULTS is set to “2.” If these attributes are not set to these values, change them as needed. You cannot change the Occurrence attribute by modifying it as a parameter. You must go to the list of elements and modify them as elements. For instructions in modifying these elements, see Modifying Elements in Chapter 3, “Specifying Business Service Contracts.”

Exporting & Testing Operations

The interface and operation that you’re creating, exporting, and testing will become the palette and template for your process flow. The process flow directs and drives your entire eLink system.

8. Export the operations within the SIMPLOPS interface if they are not already exported. See the Export Instructions in Chapter 3, “Specifying Business Service Contracts,” for instructions.
9. Export the SimpleTeller operation. See the Export Instructions in Chapter 3, “Specifying Business Service Contracts,” for instructions.
10. Test the SubtractInteger and AddInteger operations using the test button. For instructions, see Test Operation Instructions in Chapter 3, “Specifying Business Service Contracts.”

Generating Palette & Template

In this series of steps, you will generate the palette and template that will be used to form your process flow. You must use only the generated palette and template to form your process flow because only they have the necessary components to form a proper eLink process flow.

11. Select the newly created SIMPLOPS interface. then click on the Palette button.
The Palette Generation Results dialog appears. The log file displays in the dialog indicating the actions performed to generate the Palette from the selected interface. For instructions, see Generating a Palette in Chapter 4, “Designing Business Processes.”

The Palette Generation Results window appears displaying a list of generation actions.

12. Wait until the log is done displaying, then select the OK button.
13. Select operation SimpleTeller, then the Template button.

The Template Generation Results window appears displaying a list of generation actions.

14. Wait until the log is done displaying, then select the OK button.

Moving to the Business Process Window

The Business Process Window is the part of the eLink Business Process Option where you form your process flow. Be sure to use these instructions to move to the Business Process Window. Using these instructions ensures the correct configuration of your Business Process Window.

15. Click on the Edit Flow button to start the Business Process Window.

A login dialog appears.

16. Click on the Login button to confirm your login to the Business Process database.
There is no need to change any of the entries in the login dialog.

The Business Process Window appears.

Setting Up Your Process Flow

The Business Process Window allows you to form process flows from generated palettes and templates.

Note: The only process flows that will work in eLink are those that are formed from generated palettes and templates.

The first step in forming your process flows is to copy the tasks, which will be the elements of your process flow, from your palettes to your templates.

17. Click on the Open Process button.

The Process Query dialog appears.

18. Select **All eLink Palettes** from the left window, then the Search button.

A list of available palettes appears.

19. Select the newly created Palette: SIMPLOPS, then the OK button.

Palette: SIMPLOPS appears in a Design Pad window.

20. Click on Open Process button.

The Process Query dialog appears.

21. Select All eLink Templates, then the Search button.

Available templates display in the right window.

22. Select the SimpleTeller template generated in the Process Design Assistant, then the OK button.

The SimpleTeller template window appears in a Design Pad window. The task pane of the template should be empty the first time you use it.

23. Select the SubtractInteger icon from the Palette window, then choose **Edit** —> **Copy**.

24. Activate the SimpleTeller Template window by clicking in it, then choose **Edit** —> **Paste** from the menu bar.

This places the SubtractInteger object on the SimpleTeller Template's design pad. As an alternative to Steps 23 and 24, you can drag the SubtractInteger icon from the palette's design pad to the template's design pad. For instructions, see

Copying tasks between Design Pad windows in Chapter 4, “Designing Business Processes.”

25. Repeat steps 23 and 24 twice but use the AddInteger icon instead.

This places two AddInteger tasks on the SimpleTeller template.

Modifying Tasks and Processes

Tasks are generated from operations. Operations become tasks during palette generation when they are given default parameter assignments. In the following steps, the tasks are improved by the addition of perform conditions, dependencies, and the modification of parameter assignments.

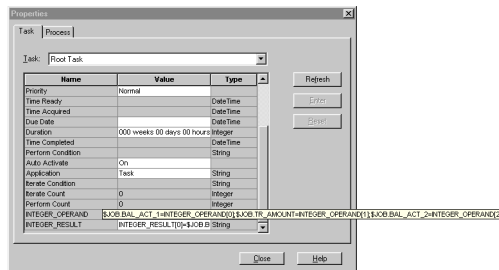
26. Add a dependency from the SubtractInteger task to one of the AddInteger tasks. For instructions, see Adding dependencies in Chapter 4, “Designing Business Processes.”

27. Add another dependency from the SubtractInteger task to the other AddInteger task.

28. Modify the root level INTEGER_OPERAND attribute value. Its attribute value should be specified as follows (For more information on modifying attribute values, see Instructions for Setting an Attribute Value or Modifying an Existing Value in Chapter 4, “Designing Business Processes.”):

```
$JOB.BAL_ACT_1=INTEGER_OPERAND[ 0 ] ;  
$JOB.TR_AMOUNT=INTEGER_OPERAND[ 1 ] ;  
$JOB.BAL_ACT_2=INTEGER_OPERAND[ 2 ]
```

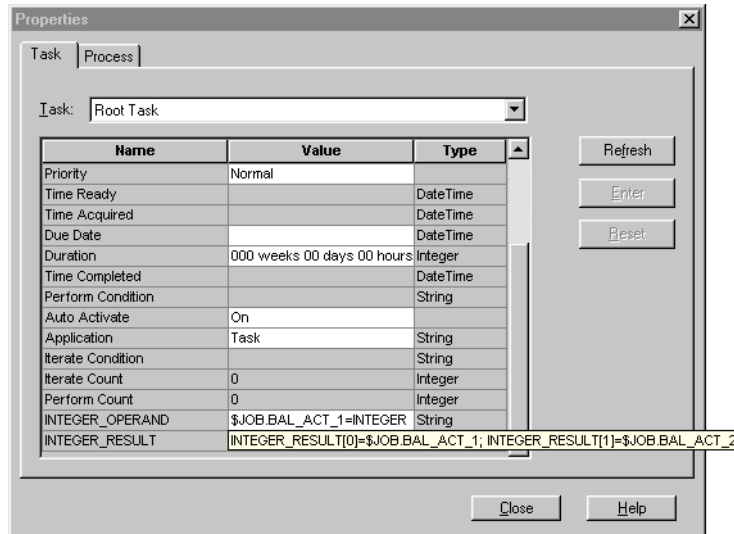
Figure B-4 Modifying Root Level INTEGER_OPERAND



29. Modify the root level INTEGER_RESULT attribute value. Its attribute value should be specified as follows:

```
INTEGER_RESULT[ 0 ]=$JOB.BAL_ACT_1 ;
INTEGER_RESULT[ 1 ]=$JOB.BAL_ACT_2
```

Figure B-5 Modifying Root Level INTEGER_RESULT



30. Modify the SubtractInteger INTEGER_OPERAND attribute value as follows:

```
INTEGER_OPERAND[ 0 ]=$JOB.BAL_ACT_1 ;
INTEGER_OPERAND[ 1 ]=$JOB.TR_AMOUNT
```

31. Modify the SubtractInteger INTEGER_RESULT attribute value as follows:

```
$JOB.BAL_ACT_1=INTEGER_RESULT[ 0 ]
```

32. Modify the first occurrence of the AddInteger INTEGER_OPERAND attribute value as follows:

```
INTEGER_OPERAND[ 0 ]=$JOB.BAL_ACT_2 ;
INTEGER_OPERAND[ 1 ]=$JOB.TR_AMOUNT
```

33. Modify the first occurrence of the AddInteger INTEGER_RESULT attribute value as follows:

```
$JOB.BAL_ACT_2=INTEGER_RESULT[ 0 ]
```

34. Modify the second occurrence of the AddInteger INTEGER_OPERAND attribute value as follows:

```
INTEGER_OPERAND[ 0 ]=$JOB.BAL_ACT_1 ;  
INTEGER_OPERAND[ 1 ]=$JOB.TR_AMOUNT
```

35. Modify the second occurrence of the AddInteger INTEGER_RESULT attribute value as follows:

```
$JOB.BAL_ACT_1=INTEGER_RESULT[ 0 ]
```

36. Add a new attribute BAL_ACT_1 with data type Integer to the process level of the SimpleTeller process. For instructions, see Instructions for Adding an Attribute to be Used in a Perform Condition in Chapter 4, “Designing Business Processes.”

37. Add a perform condition to the first occurrence of your AddInteger task, so the integers are added only if the result of SubtractInteger is greater than or equal to zero. The effect of this perform condition is that the transfer amount will only be added to the second account balance if the first account is not overdrawn by the SubtractInteger task. For instructions on adding a PerformCondition, see Adding PerformConditions to tasks in Chapter 5, “Making eLink Processes Work.” Here is the syntax of the Perform Condition to be added:

```
$JOB.BAL_ACT_1>=0
```

38. Add a perform condition to the other AddInteger tasks, so the integers are added only if the result of SubtractInteger is less than zero. The effect of this perform condition is that the transfer amount will be added back into the first account balance (hence, reversing the withdrawal) if the first account is overdrawn by the SubtractInteger (withdrawal) task. For instructions on adding a PerformCondition, see Adding PerformConditions to tasks in Chapter 5, “Making eLink Processes Work.” Here is the syntax of the Perform Condition to be added:

```
$JOB.BAL_ACT_1<0
```

Adding Your New Process to the Configuration

This series of steps adds your new process to your eLink Business Process Option configuration, so it can be run by the process runner or starter processes. To do this you must edit your eLink_BPO.cfg by adding the process name, which is also called a service name, to a SERVICE_LIST setting.

39. Logon to your Business Process Option server as the Business Process Option administrator. Your server was installed so that “icadmin” is both your user name and password. Your password has probably been changed since installation, so check with your System Administrator to determine the administrator user name and password.
40. Change directories to the directory where the eLink Business Process Option is installed. The default path for this directory is

```
/work1/ic/appdir
```

41. Edit the file eLink_BPO.cfg by adding your new process name (in this case, it’s SimpleTeller) to the SERVICE_LIST setting under either the [Server=eProcRunner] setting group (suggested) or the [Server=eProcStarter] setting group.
42. Save eLink_BPO.cfg and exit your UNIX text editor.
43. Shutdown the server upon which you are changing settings with the command

```
tmshutdown -s <server name>
```
44. Start up the server upon which you changed settings with the command

```
tmboot -s <server name>
```

Testing Your Process Flow

Your process flow is tested in the same manner as testing any operation in the Business Interface Window. However, the connected applications, the eLink Platform, the Data Integration Option, and the Business Process Option can all come into play when testing a process flow.

45. Exit from the Business Process Window, then select the SimpleTeller Operation.
46. Click on the Test button.
47. Enter data in the data fields, then click OK.

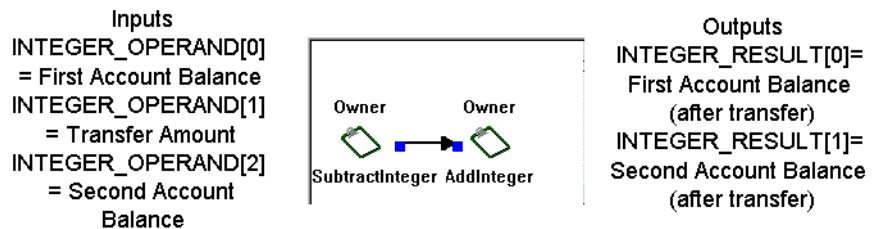
Assigning Parameters to Implement Process Flows

This section is provided to explain the mechanics of the funds transfer tutorial from the previous section in more detail as the more high-level conceptual information is presented here.

Root Level Input & Output

Figure B-6 shows inputs and outputs to and from the SimpleTeller process flow. This process flow implements a simple funds transfer. It is considered a simplified transfer because only whole dollar amounts may be transferred (parameters are of integer type) and because addition and subtraction tasks are used in place of actual deposit and withdrawal business services.

Figure B-6 SimpleTeller Process Flow Inputs & Outputs



We begin by looking at the top level of the process definition. This level of the process is also known as the Root Task. The process must “know” the input and output information presented in Figure B-6 to execute the process correctly. The input information coming into the process from the caller (INTEGER_OPERAND[*]) must be assigned to job-level attributes, and the output job-level attributes must be assigned to the output going out of the process as a reply to the caller. To do this (for the purposes of the funds transfer example), the following must be assigned as the value of the root-level INTEGER_OPERAND attribute:

```
$JOB.BAL_ACT_1=INTEGER_OPERAND[0];  
$JOB.TR_AMOUNT=INTEGER_OPERAND[1];  
$JOB.BAL_ACT_2=INTEGER_OPERAND[2]
```

and, the following must be assigned as the value of the root-level INTEGER_RESULT attribute:

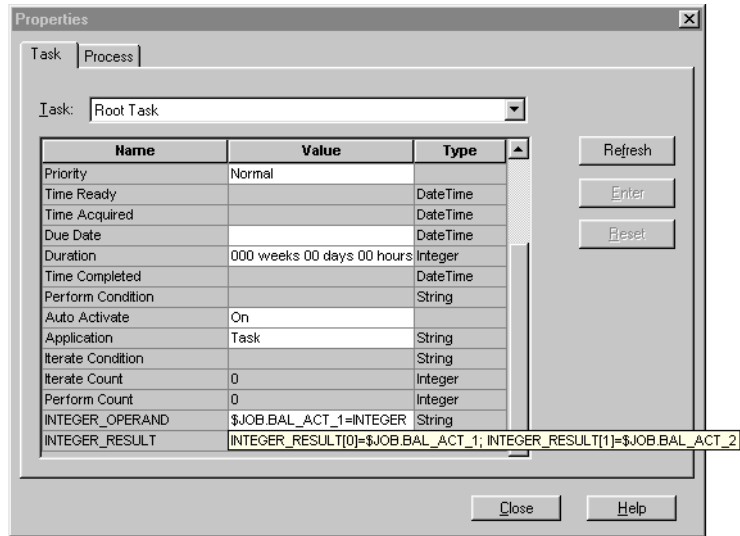
```
INTEGER_RESULT[0]=$JOB.BAL_ACT_1;
INTEGER_RESULT[1]=$JOB.BAL_ACT_2
```

Assigning these values to the root-level parameter attributes defines the inputs and outputs to and from the process at the process (root) level.

The job-level attribute names BAL_ACT_1, TR_AMOUNT, and BAL_ACT_2 were arbitrarily created and assigned and each of them can be any string of alpha-numeric (UPPERCASE) characters. It is suggested that they describe the function they are to perform. For example, BAL_ACT_1 is short for Balance of Account 1.

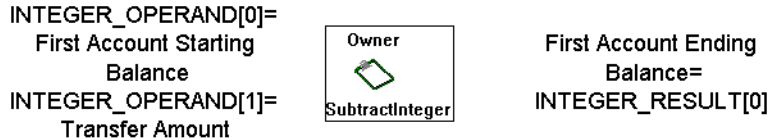
Figure B-7 shows the Root Task properties screen with parameter assignments made as specified above.

Figure B-7 INTEGER_RESULT Root-Level Parameter Assignments



Task Level Input & Output (SubtractInteger)

Figure B-8 shows inputs and outputs to and from the SubtractInteger task. This task subtracts one integer from another. It is used to simulate a withdrawal of funds from a bank account.

Figure B-8 SubtractInteger Task Inputs & Outputs

SubtractInteger is a sub-task of the root task, so the job-level attributes created in the Root Task can be used. These are the expressions that should be entered as the value for the SubtractInteger INTEGER_OPERAND attribute:

```
INTEGER_OPERAND[0]=$JOB.BAL_ACT_1;  
INTEGER_OPERAND[1]=$JOB.TR_AMOUNT
```

and the expression entered as the value for the SubtractInteger INTEGER_RESULT attribute is:

```
$JOB.BAL_ACT_1=INTEGER_RESULT[0]
```

The actual operation performed at the application level is:

```
INTEGER_RESULT[0]=INTEGER_OPERAND[0]-INTEGER_OPERAND[1]
```

If you use substitution, the equivalent expression is:

```
$JOB.BAL_ACT_1=$JOB.BAL_ACT_1-$JOB.TR_AMOUNT
```

So, looking at this equation, you can see that a new balance is obtained for Account 1 by subtracting the transfer amount from Account 1's starting balance. This is the action that occurs in the withdrawal portion of a funds transfer. Notice that the same job-level attribute is used for the starting and ending account balance.

Task Level Input & Output (AddInteger)

Figure B-9 shows inputs and outputs to and from the AddInteger task. This task adds one integer to another. It is used to simulate a deposit of funds to a bank account.

Figure B-9 AddInteger Task Inputs & Outputs



AddInteger is a sub-task of the root task, so the job-level attributes created in the Root Task can be used. These are the expressions that should be entered as the value for the AddInteger INTEGER_OPERAND attribute:

```
INTEGER_OPERAND[0]=$JOB.BAL_ACT_2;  
INTEGER_OPERAND[1]=$JOB.TR_AMOUNT
```

and the expression entered as the value for the SubtractInteger INTEGER_RESULT attribute is:

```
$JOB.BAL_ACT_2=INTEGER_RESULT[0]
```

The actual operation performed at the application level is:

```
INTEGER_RESULT[0]=INTEGER_OPERAND[0]+INTEGER_OPERAND[1]
```

If you use substitution, the equivalent expression is:

```
$JOB.BAL_ACT_2=$JOB.BAL_ACT_2+$JOB.TR_AMOUNT
```

So, looking at this equation, you can see that a new balance is obtained for Account 2 by adding the transfer amount to Account 2's starting balance. This is the action that occurs in the deposit portion of a funds transfer. Notice that the same job-level attribute is used for the starting and ending account balance.

MATHAPP Service Reference

Table B-1 lists all of the MATHAPP services and provides a description of the required inputs and outputs. These services can be included in operations, then in interfaces in the Business Interface Window. However, you must import mathserv.fml before doing so.

Table B-1 MATHAPP Service Name Parameters: Input and Output

Service Name	Input Type	FML Field Name	Occurrence	Input/Output
AddInteger	long	INTEGER_OPERAND	0	Input
	long	INTEGER_OPERAND	1	Input
	long	INTEGER_RESULT	0	Output
AddFloat	float	FLOAT_OPERAND	0	Input
	float	FLOAT_OPERAND	1	Input
	float	FLOAT_RESULT	0	Output
SubtractInteger	long	INTEGER_OPERAND	0	Input
	long	INTEGER_OPERAND	1	Input
	long	INTEGER_RESULT	0	Output
SubtractFloat	float	FLOAT_OPERAND	0	Input

Table B-1 MATHAPP Service Name Parameters: Input and Output

Service Name	Input Type	FML Field Name	Occurrence	Input/Output
	float	FLOAT_OPERAND	1	Input
SubtractFloat	float	FLOAT_RESULT	0	Output
MultiplyInteger	long	INTEGER_OPERAND	0	Input
	long	INTEGER_OPERAND	1	Input
	long	INTEGER_RESULT	0	Output
MultiplyFloat	float	FLOAT_OPERAND	0	Input
	float	FLOAT_OPERAND	1	Input
	float	FLOAT_RESULT	0	Output
DivideInteger	long	INTEGER_OPERAND	0	Input
	long	INTEGER_OPERAND	1	Input
	long	INTEGER_RESULT	0	Output
DivideFloat	float	FLOAT_OPERAND	0	Input
	float	FLOAT_OPERAND	1	Input
	float	FLOAT_RESULT	0	Output

Table B-1 MATHAPP Service Name Parameters: Input and Output

Service Name	Input Type	FML Field Name	Occurrence	Input/Output
SquareInteger	long	INTEGER_OPERAND	0	Input
	long	INTEGER_RESULT	0	Output
SquareFloat	float	FLOAT_OPERAND	0	Input
	float	FLOAT_RESULT	0	Output
SumInteger	long	INTEGER_OPERAND	0-n	Input
	long	INTEGER_RESULT	0	Output
SumFloat	float	FLOAT_OPERAND	0-n	Input
	float	FLOAT_RESULT	0	Output
SRootInteger	long	INTEGER_OPERAND	0	Input
	long	INTEGER_RESULT	0	Output
SRootFloat	float	FLOAT_OPERAND	0	Input
	float	FLOAT_RESULT	0	Output
TimeInterval	char *	DATE_TIME	0	Input
	long	INTERVAL	0	Output
CurrencyFmt	float	AMOUNT	0	Input

Table B-1 MATHAPP Service Name Parameters: Input and Output

Service Name	Input Type	FML Field Name	Occurrence	Input/Output
CurrencyFmt	char *	FORMAT_MASK	0	Input
	char *	STRING_AMOUNT	0	Output
Compound	float	CAPITAL	0	Input
	float	INTEREST_RATE	0	Input
	long	TERM_YEARS	0	Input
	float	COMPOUND_AMOUNT	0-TERM_YEARS	Output

Glossary

activated state

Task state when a task with subtasks is started automatically. A task enters this state when it is ready to do and its AutoActivate property is on. An activated task is also completed automatically.

active process

A copy of a process definition created when the process is started. An active process is also called a *process instance* because you can create many active processes from the same process definition.

admin pool

A special pool for users who need Administrator privilege. During process design, you need Administrator privilege to create new attributes and process classes.

API

Application Programming Interface

Application to Transaction Monitor Interface

This is the set of routines that make up the BEA TUXEDO communications library. Programs that can call the ATMI are TUXEDO clients.

asynchronous communication paradigm

A communication in which the requestor does not wait for the response but will poll for it at a later time.

ATMI

See Application to Transaction Monitor Interface.

attribute

Information about a process or task. Some attributes are built in, such as Due Date or Duration. Others are user-defined, such as a customer name or approval code. See also *process attribute* and *task attribute*.

attributes window

Activity window where you add attributes to the process and set attribute values.

AutoActivate

Attribute that determines if a task is started and completed automatically. By default, AutoActivate is on for all tasks in a process. Also, eLink won't work properly if AutoActivate is turned off.

Base Job Template

A predefined process definition consisting of the root task only. Your first process definition must be built from a copy of the Base Job Template. You can create as many additional processes from the Base Job Template as you like.

buffer type

Messages are stored in queues as described previously. These messages are stored in buffers that are of the five listed buffer types. The only buffer type allowed in eLink is FML32.

Business Process

A set of one or more linked procedures or activities which collectively realize a business objective or policy goal, normally within the context of an organizational structure defining functional roles and relationships.

Business Process window

Window displayed when you confirm your login after clicking on the Edit Flow button. Contains a menu bar, tool bar, status bar and work area where other windows are displayed.

child task

A subtask of another task.

class

A category for a process definition or other object. During process design, you can create new process classes, but you can only use existing document classes. Document classes can be created only with the Batch Registry utility.

conditional workflow

A process definition where tasks are performed only under certain circumstances. See also *PerformCondition*.

contract

a contract is an agreement between a eLink and the application about the inputs and expected outputs of operations.

Contract Repository

a database that stores the contracts created in the Process Design Assistant Business Interface Window.

data type

refers to the type of data. In this book, these are the standard C data types (int, double, long, string, etc.)

design pad

Activity window where you lay out and view the process structure.

eLink Platform

A focused usage of BEA TUXEDO that supports eLink application integration. The Business Process Option requires the eLink Platform as prerequisite software.

explicit dependency

User-defined dependency between sibling tasks. In a process definition, you indicate an explicit dependency by drawing an arrow from the precedent task to the dependent task. The dependent task cannot be started until the precedent task is done.

export

to make operations or interfaces available for testing, then for forming into process flows.

Field Manipulation Language (FML)

generically, an interface for maintaining buffers with field/value pairs. In TUXEDO, this refers specifically to the 16-bit version. The 32-bit version is called FML32.

flat process definition

A process definition where all tasks with subtasks start and complete automatically. The AutoActivate property is on for all tasks in the process, effectively creating a single-level task hierarchy. This is the default process model.

FML

See **Field Manipulation Language (FML)**

FML32

See **Field Manipulation Language (FML)**

hierarchy pane

The left side of the Design Pad. Displays the structure of the process as a whole, similar to the way the Windows File Manager displays the directory structure of the Windows file system. *See also* *task pane*.

implicit dependency

Built-in dependency of a task on its parent. A subtask cannot be started until its parent is started.

interface

A shared boundary between eLink and the application connected to it. It is defined by the inputs from eLink to the application and the outputs from the application to eLink.

module

a grouping of interfaces.

operation

a business calculation performed in an eLink application. It requires inputs and most times yields an expected output. In the eLink Business Process Option, the specified inputs and expected outputs are referred to as an operation.

owner pool

Pool assigned to the built-in Owner role. You become the process owner by being a member of the owner pool.

Owner role

A built-in role for the root task. Before any role assignments are made, every task in a process has the Owner role.

parent task

A task with subtasks.

PDA

See Process Design Assistant (PDA).

PerformCondition

A condition to be tested when a task becomes ready to do. InConcert can then decide whether to skip the task or let it be performed. A PerformCondition consists of attributes, values, and operators (+ – AND, and so on).

pool

A group of users who perform the same job. Each user must belong to at least one pool; typically, users belong to several pools.

privilege

An access control mechanism for processes. There are four kinds of privilege: Manage, Update, Copy, and Read.

process attribute

Attribute that applies to the process as a whole. When you set a process attribute in one task, its value is the same in every other task. See also *task attribute*.

process definition

The representation of a business process in a form which supports automated manipulation, such as modeling, or enactment by a workflow management system. The process definition consists of a network of activities and their relationships, criteria to indicate the start and termination of the process, and information about the individual activities, such as participants, associated IT applications and data, etc.

A general description of a business activity, which automates the flow of tasks and other information among members of a work group who are collaborating on a project.

Process Definition Group

An administrative classification of related process definitions within a workflow management system.

Process Design Assistant (PDA)

The graphical toolset for designing process definitions and their component activities. (The Process Design Assistant uses a repository that stores data about interface contracts between clients and servers.)

process instance

See *active process*.

process owner

Person responsible for managing the process. You must be a member of the owner pool to be the process owner.

property

Auxiliary information about an object in the database, such as a process or task. Most properties appear as view-only “attributes” in the Attributes window. Some properties are modifiable, such as role, pool, and AutoActivate.

query

A search for information in the database. There are two kinds of query: process query and document query.

repository tree

the hierarchical structure that displays the contracts and their component parts.

role

Description of the skills needed to perform a task; for example, manager, clerk, or buyer. Each task in a process has one role.

root task

The parent of all other tasks in a process. All processes have a root task.

server

eLink software that controls and coordinates all Business Process functions. Each time you start the eLink Business Process Option, you actually log in to a Business Process server with your user name and password.

sibling tasks

Subtasks with the same parent.

status bar

Bottom of the Business Process window where messages and abbreviated help text are displayed.

struct

groups of elements. The elements can be from any of the five allowable data types.

suspended process

Process in which work has been halted temporarily. No work can be performed on the tasks in a suspended process unless the work has already begun.

synchronous communication paradigm

A communication in which the requestor waits for the reply.

task

A unit of work in a process that is performed by one *worker*.

task attribute

Attribute that applies to a single task in a process. For example, each task can have a different value for the Due Date attribute. See also *process attribute*.

task pane

Right side of the Design Pad. When you select a task in the Hierarchy (left) pane, its child tasks, if any, are displayed in the Task pane. See also *hierarchy pane*.

task state

Status of a task. The task state determines if work can be performed on the task. A task can have one of six states: Waiting, Ready, Bypassed (or Skipped), Activated, In Process (or Acquired), or Done. See also *activated state* and *AutoActivate*.

toolbar

The buttons at the top of the Business Process window. The left side of the toolbar provides access to activities. The right side of the toolbar provides access to individual commands available in the particular activity (Process Designer, Process Manager, and so on) you are currently using.

transaction

A collection of tasks treated as a single unit of work within ATMI. The main value of transactions is that they ensure the atomicity of a set of operations that together form a business rule.

user

A person that starts, works on, and completes tasks in a process. Each user has a user name and password defined in the Business Process database.

worker

software agent or user to which tasks are assigned.

Index

A

- activated state 2-5
- active process
 - owner 2-12
 - task states 2-9
- add a dependency 4-13
- add a PerformCondition 4-13
- add an IterateCondition 4-13
- Add Module window 3-33
- add object
 - menu, keyboard shortcut 3-11
- Adding elements
 - instructions 3-24
- adding elements
 - to Process Design Assistant 3-22
- adding interfaces
 - instructions for Contract Repository 3-32
 - to Process Design Assistant 3-31
- adding IterateConditions to tasks 5-23
- adding modules
 - in Process Design Assistant 3-33
 - instructions for Contract Repository 3-34
- adding objects
 - with Process Design Assistant 3-22
- adding operations
 - instructions 3-31
 - to Process Design Assistant 3-28
- adding PerformConditions to tasks 5-17
- Adding struct
 - instructions 3-28
- adding structures
 - to Process Design Assistant 3-26
- alert window
 - instructions 3-56
- aligning tasks 4-27, 4-28
- Alignment Grid 4-28
- alignment grid 4-28
 - hiding 4-28
 - showing 4-28
- attributes
 - AutoActivate 2-5
 - built-in 2-10, 2-12
 - changing display 4-26
 - data types 2-12
 - defined 2-10
 - in PerformConditions 2-7, 5-9
 - internal for processes 4-22
 - internal for tasks 5-14
 - modifying 4-35
 - Name 4-22
 - PerformCondition data types 5-11
 - prefixes in PerformConditions 5-9
 - process 2-12
 - Process Creator 4-22
 - resetting 4-35
 - setting values 4-35
 - specifying 4-34
 - specifying values to start process 5-15
 - task 2-12
 - Time Created 4-22

user-defined 2-10, 2-12
AutoActivate property 2-5, 2-10

B

Batch Registry
 listing attribute names with 5-13
built-in attributes 2-10
bulk load file
 description A-2
 loading BULKPKG A-2
 syntax A-2
bulk loader
 with the Contract Repository A-2
bulk loader file
 keyword order A-3
 example A-4
bulk loader utility
 troubleshooting A-6
bulk loading services
 using Contract Repository Editor A-2

C

case-sensitivity
 in attribute names 5-9
 in PerformCondition strings 5-11
closing Design Pad 4-11
completing
 process 5-32
compound statements 5-12
conditional process definition 2-6
conditional workflow 2-6
Contract Repository
 alert window 3-56
 bulk loads A-2
 Delete Module window 3-48
 deleting objects 3-48
 editing services A-2
 exit prompt 3-55
 Exported operation window 3-51

exporting an operation 3-50, 4-2, 4-4, 4-5
guidelines for inputting data 3-54
in depth 3-12
Logoff 3-55
Logon options 3-15
requirements 3-13
test operation instructions 3-54
test operation parameter
 Cancel 3-54
 Clear 3-54
 Input/Output Buffer 3-53
 Operation 3-53
 Parameter text field 3-54
 Run 3-54
 TUXEDO Service 3-53
Test Operation window 3-53
testing operations 3-53
troubleshooting 3-56
unexporting an operation 3-52

Contract Repository Editor
 Element Selection window 3-35
 Interface Selection window 3-44
 Modify Element window 3-36
 Modify Interface window 3-45
 Modify Module window 3-47
 Modify Operation window 3-43
 modify parameter option
 Display Name 3-41
 Exception Mode 3-41
 Mandatory 3-41
 Occurrence 3-40
 Parameter Name 3-40
 Passing Mode 3-41
 Return Value 3-40
 Type 3-40
 Modify Parameter window 3-40
 Modify Struct Parameter window 3-38
 modifying elements 3-35
 modifying interfaces 3-43
 modifying modules 3-45

- modifying objects 3-34
- modifying operations 3-41
- modifying parameters 3-38
- modifying structures 3-37
- module option
 - arrow keys 3-34
 - Available Interfaces 3-33
 - Cancel 3-34
 - Help 3-34
 - Last Updated 3-34
 - OK 3-34
- Modules Selection window 3-46
- Operation Selection window 3-42
- Parameter Selection window 3-39
- Structure Selection window 3-37
- copy a task 4-12
- copying tasks 4-17, 4-22
 - with drag and drop 4-23
- create object
 - menu, keyboard shortcut 3-11
- creating
 - folders in Process Manager 5-39

D

- database
 - description 3-3
- default system icons 4-33
- delete a dependency 4-13
- delete a PerformCondition 4-13
- delete a task 4-13
- delete an IterateCondition 4-13
- delete object
 - menu, keyboard shortcut 3-11
- Delete Process dialog box 5-33
- deleting
 - completed process 5-33
- deleting dependencies 4-30
- deleting objects
 - from Contract Repository 3-48
- instructions 3-48

- deleting tasks 4-30
- dependencies 2-8
 - adding 4-29
- dependent task 2-8
- deselecting a task 4-25
- Design Pad
 - aligning tasks 4-27
 - closing 4-11
 - copying tasks between multiple windows 4-17
 - displaying attributes 4-26
 - drag and drop 4-17
 - hiding child tasks in Hierarchy pane 4-15
 - Hierarchy pane 4-9
 - navigating from 4-14
 - layout operations 4-12
 - multiple 4-16
 - multiple windows 4-16
 - opening 4-8
 - parts of 4-8
 - printing 4-11
 - refreshing 4-10
 - showing subtasks in hierarchy pane 4-14
 - Task pane 4-9
 - navigating from 4-15

- Design Pad window 4-9

- documents

- moving between Document Organizer folders 5-41

- drag and drop 5-41

- between Design Pads 4-17
 - copying tasks with 4-23

E

- editing task names 4-27
- element
 - creating 3-22
 - deleting 3-48
 - modifying 3-35
- exit

- menu, keyboard shortcut 3-10, 3-11
- expanded task icon 4-14
- explicit dependencies 2-8
 - between non-sibling tasks 2-8
- export an operation
 - menu, keyboard shortcut 3-11
- exporting
 - instructions 3-51, 4-3

F

- flat process definitions 2-5
- FML files
 - import menu, keyboard shortcut 3-11
 - importing 3-17
- FML/VIEW import
 - instructions 3-20
- folders
 - creating in Process Manager 5-39
 - finding and adding process definitions to 5-39
 - moving documents between 5-41
 - Process Manager 5-31
 - using with Process Manager 5-38

G

- gray lines, in Design Pad 4-28
- grid 4-28

H

- Hierarchy pane 4-14

I

- icon library 4-31
- icons
 - for process 5-31
- implicit task dependencies 2-8
- import FML file
 - menu, keyboard shortcut 3-11

- import VIEW file
 - menu, keyboard shortcut 3-11
- import workflow
 - FML/View 3-19
- importing files
 - existing TUXEDO files 3-17
 - FML 3-17
 - VIEW 3-17
- interface
 - creating 3-31
 - deleting 3-48
 - modifying 3-43
- internal process attributes 4-22
- internal task attributes 5-14
- Iterate Condition Wizard 5-23
 - Combining Expressions 5-26
 - Creating Expressions 5-25
 - Ordering Evaluations 5-27, 5-28
 - Selecting Attributes 5-24
- IterateCondition 5-8
 - add 4-13
 - delete 4-13
 - specifying 5-23
 - specifying based on task attributes 5-24
 - specifying with expressions 5-28
 - See also* PerformCondition
- iterating subtasks 5-16

J

- \$JOB prefix 5-9

K

- keyboard shortcuts 3-10
- keywords
 - CURRENT 5-12
 - for service definitions A-3
 - parameter level A-5
 - service level A-4

L

- logical operators 2-7
- Logoff
 - instructions 3-55
- logoff
 - menu, keyboard shortcut 3-11
- logon
 - menu, keyboard shortcut 3-10
- logon default settings
 - menu, keyboard shortcut 3-12
- Logon options
 - for Contract Repository 3-12, 3-15

M

- main window
 - refresh display 3-8
- menu shortcuts 3-10
- modify object
 - menu, keyboard shortcut 3-11
- Modify Process dialog box 5-34
- modifying
 - task assignments with Process Manager 5-33
- modifying elements
 - Contract Repository Editor 3-35
 - instructions 3-36
- Modifying Interface
 - instructions 3-44
- modifying interfaces
 - in Contract Repository Editor 3-43
- modifying modules
 - in Contract Repository Editor 3-45
 - instructions 3-46
- modifying modules See also Adding Modules
- modifying objects
 - in Contract Repository Editor 3-34
- modifying operations
 - in Contract Repository Editor 3-41
 - instructions 3-42

- modifying parameter
 - instructions 3-39
- modifying parameters
 - in Contract Repository Editor 3-38
- modifying structs
 - instructions 3-37
- modifying structures
 - in Contract Repository Editor 3-37
- module
 - creating 3-33
 - deleting 3-48
 - modifying 3-45
- move a task 4-12
- moving a task 4-25

N

- Name attribute 4-22
- navigating from the Property Sheet 4-15
- navigating the process structure 4-13
- New Process Folder dialog box 5-39

O

- object tree 3-4
- online help
 - F1 key command 3-12
- operation
 - creating 3-28
 - deleting 3-48
 - exporting 3-50, 4-2, 4-4, 4-5
 - modifying 3-41
 - testing 3-53
 - unexporting 3-52
- owner pool 2-12

P

- parameter
 - creating See *adding an element* 3-22
 - deleting 3-48

-
- modifying 3-38
 - parameter keyword
 - access A-5
 - count A-5
 - param A-5
 - type A-5
 - parent task
 - using for organization 2-5
 - Perform Condition Wizard 5-17
 - Combining Expressions 5-20
 - Creating Expressions 5-20
 - Ordering Evaluations 5-21, 5-22
 - Selecting Attributes 5-18
 - PerformCondition 5-8
 - add to task 4-13
 - attribute prefixes 5-9
 - attributes 5-9
 - compound statements in 5-12
 - CURRENT keyword 5-12
 - data types 5-11
 - defined 2-6
 - delete 4-13
 - evaluation order 5-12
 - example 2-7
 - examples 5-16
 - format 5-9
 - maximum length 5-9
 - operators 2-7, 5-11
 - properties 5-10
 - simulating branching with 2-7
 - specifying 5-17
 - specifying based on task attributes 5-18
 - specifying built-in attributes 5-13
 - specifying with expressions 5-22
 - statements 5-9
 - precedent task 2-8
 - printing from the Design Pad 4-11
 - process
 - complete, force 5-32
 - delete complete 5-33
 - icons 5-31
 - modifying task assignments 5-33
 - resuming 5-34
 - specifying attribute values to start 5-15
 - suspending 5-34
 - viewing status with Process Manager 5-35
 - process attributes 2-12
 - internal 4-22
 - Process Creator attribute 4-22
 - Process Definition
 - conditional 2-6
 - flat 2-5
 - kinds of 2-5
 - modifying 4-19
 - saving 4-11
 - saving with new name 4-19
 - process definition
 - design and layout operations 4-12
 - finding and adding to Process Manager folder 5-39
 - Process Definition window 4-15
 - Process Design Assistant
 - add element option
 - Cancel 3-24
 - Comments 3-24
 - Element Length 3-23
 - Element Name 3-23
 - Element Type 3-23
 - Fixed 3-24
 - FML Field Name 3-24
 - FML Field Number 3-24
 - Help 3-24
 - Last Updated 3-24
 - Occurrence 3-24
 - OK 3-24
 - Add Element window 3-23
 - add interface option
 - arrow keys 3-32
 - Available Operations 3-32
 - Cancel 3-32
 - Current Operations 3-32

- Help 3-32
- Interface Name 3-32
- Last Updated 3-32
- OK 3-32
- Add Interface window 3-31
- add operation option
 - Available Parameters 3-30
 - Cancel 3-30
 - Current Parameters 3-30
 - Edit 3-30
 - Exported 3-30
 - Help 3-30
 - Input/Output Buffer type 3-30
 - Last Updated 3-30
 - OK 3-30
 - Operation Name 3-29
 - Return Exception 3-30
 - TUXEDO Service Name 3-30
- Add Operation window 3-29
- add struct option
 - arrow keys 3-27
 - Available Members 3-27
 - Cancel 3-28
 - Current Members 3-27
 - Help 3-28
 - Last Updated 3-27
 - OK 3-27
 - Struct Name 3-27
- Add Struct window 3-27
- adding elements 3-22
- adding interfaces 3-31
- adding modules 3-33
- adding objects 3-22
- adding operations 3-28
- adding structures 3-26
- description 3-3
- logging on 3-15
- module option
 - Current Interfaces 3-33
 - Module name 3-33
- overview 3-1
 - starting 3-14
- Process Designer
 - toolbar buttons
 - tool bar buttons 4-7
- Process Designer window 4-28
- Process Manager
 - creating new folders 5-39
 - finding and adding process definitions to folders 5-39
 - folders 5-31
 - opening 5-30
 - using folders 5-38
- Process Manager window 5-31
- process owner 2-12
- Process Query dialog box 4-19, 4-20, 5-40
- Process Status window 5-36
- process structure
 - defining 4-12
 - navigating 4-13
- processes
 - activated 2-5
 - attributes 2-12
 - changing structure 4-12
 - defining structure 4-12
 - design and layout operations 4-12
 - internal attributes 4-22
 - navigating from Hierarchy pane of Design Pad 4-14
 - navigating from Property Sheet 4-15
 - navigating structure 4-13
 - navigating with Task pane in Design Pad 4-15
 - owner pool 2-12
 - printing 4-11
 - process owner 2-12
 - saving 4-11
 - task states 2-9
 - viewing in multiple windows 4-16
 - working on different parts 4-16
- properties
 - in PerformConditions 5-10

Property Sheet 4-34
 navigating from 4-15
Process page 4-21
Task page 4-34

Q

quitting Design Pad 4-11

R

refresh
 menu, keyboard shortcut 3-11
removing objects
 from Contract Repository 3-48
removing task dependencies 4-30
removing tasks 4-30
Rename Folder dialog box 5-40
renaming tasks 4-27
resetting attribute values 4-35
resume process 5-34

S

saving Process Definitions 4-11
select a task 4-12
Select an Icon dialog box 4-32
selecting and moving tasks 4-25
selecting tasks 4-25
service definition
 keyword order A-3
 keywords A-3
service keyword
 export A-4
 inbuf/outbuf A-5
 service A-4
setting Logon options
 for Contract Repository 3-12, 3-15
Show Grid command 4-28
Snap to Grid command 4-27
states of tasks 2-9

structure
 creating 3-26
 deleting 3-48
 modifying 3-37
subtasks
 in flat process definitions 2-5
subtasks, iterating 5-16
support
 technical xvi
suspend process 5-34
syntax
 bulk load data file A-2

T

\$TASK prefix 5-9
task
 add dependency 4-13
 add Iterate Condition 4-13
 add PerformCondition 4-13
 copy 4-12
 delete 4-13
 delete dependency 4-13
 delete IterateCondition 4-13
 delete PerformCondition 4-13
 move 4-12
 select 4-12
task attributes 2-12
 built-in 2-12
 internal 5-14
 modifying 4-35
 resetting values 4-35
 setting values 4-35
 specifying 4-34
 using to specify IterateConditions 5-24
 using to specify PerformConditions 5-18
Task Customize dialog box 5-43
task dependencies
 adding 4-29
 deleting 4-30
 explicit 2-8

-
- implicit 2-8
 - Task Details window 5-37
 - task icon
 - changing 4-31
 - customizing 4-30
 - default 4-33
 - display areas 4-26
 - expanded 4-14
 - icon libraries 4-31
 - system default 4-33
 - Task Icon Customize dialog box 4-27, 5-37
 - Task Icon Palette 4-31, 4-32, 4-33
 - adding icons to 4-32
 - hiding 4-31
 - removing icons from 4-33
 - Task Property Sheet 5-38
 - task states 2-9
 - Acquired 2-10
 - Activated 2-10
 - AutoActivate 2-10
 - Bypassed 2-10
 - Done 2-10
 - In Process 2-10
 - Ready 2-9
 - Skipped. *See* task states: Bypassed
 - Waiting 2-9
 - task structure 2-8
 - tasks
 - adding dependencies 4-29
 - adding IterateConditions 5-23
 - adding PerformConditions 5-17
 - aligning in Design Pad 4-27
 - attributes 2-12
 - built-in attributes 2-12
 - changing icons 4-31
 - copying 4-22
 - copying in Design Pad 4-17
 - copying with drag and drop 4-23
 - customizing icons 4-30
 - default icons 4-33
 - default system icons 4-33
 - deleting 4-30
 - deleting dependencies 4-30
 - dependencies 2-8
 - dependent 2-8
 - deselecting 4-25
 - displaying attributes in Design Pad 4-26
 - expanded icon 4-14
 - explicit dependencies 2-8
 - hiding child tasks in Design Pad 4-15
 - implicit dependencies 2-8
 - internal attributes 5-14
 - modify assignments in Process Manager 5-33
 - moving 4-25
 - precedent 2-8
 - removing dependencies 4-30
 - renaming 4-27
 - selecting 4-25
 - selecting and moving 4-25
 - selecting via Property Sheet 4-15
 - showing subtasks in Design Pad 4-14
 - specifying attributes 4-34
 - states 2-9
 - using parent task for organization 2-5
 - test
 - menu, keyboard shortcut 3-11
 - Time Created attribute 4-22
 - Toolbar buttons
 - Add 3-9
 - Delete 3-9
 - Export 3-9
 - Import FML/VIEW 3-9
 - Logoff 3-9
 - Logon 3-9
 - Modify 3-9
 - Settings 3-10
 - Test 3-10
 - Unexport 3-10
 - troubleshooting
 - bulk loader utility A-6
 - TUXEDO

- bulk loading services into database A-2
- TUXEDO applications
 - support for 3-2
- TUXEDO services
 - loading with bulk loader 3-2

U

- unexport an operation
 - menu, keyboard shortcut 3-11
- user-defined attributes 2-10, 2-12
- using
 - parameter level keywords A-5
 - service level keywords A-4
- using alignment grid 4-28

V

- values
 - for parameter level keywords A-5
 - for service level keywords A-4
- VIEW files
 - import menu, keyboard shortcut 3-11
 - importing 3-17
- viewing
 - process status with Process Manager 5-35

W

- window
 - Process Manager 5-31
- workflow 2-3